PROMQUEEN

The PROMQUEEN Manual

Gloucester Computer, Inc.

LIST OF ILLUSTRATIONS

LIST OF TABLES

PROPRIETARY NOTICE

The information and product design as disclosed in this manual (as they pertain to the PROMQUEEN TM Cartridge Development System and/or the HEXKIT TM software) were originated by and are the property of Gloucester Computer, Incorporated.

The contents of this manual have been reviewed for accuracy; however, no responsibility is assumed should any portion not be accurate.

Specifications and information herein are subject to change to allow the introduction of design improvements.

VIC, VIC-20, Commodore 64, and VICMON are registered trademarks of Commodore Business Machines, Inc.

1 Blackburn Center
Gloucester, Massachusetts 01930

Printed in U.S.A.

Dear New Owner:

Welcome to the third computer generation--a generation characterized by your need for more powerful, sophisticated tools with fewer complications and less cost. This is truly the generation of the independent-minded, highly competent user.

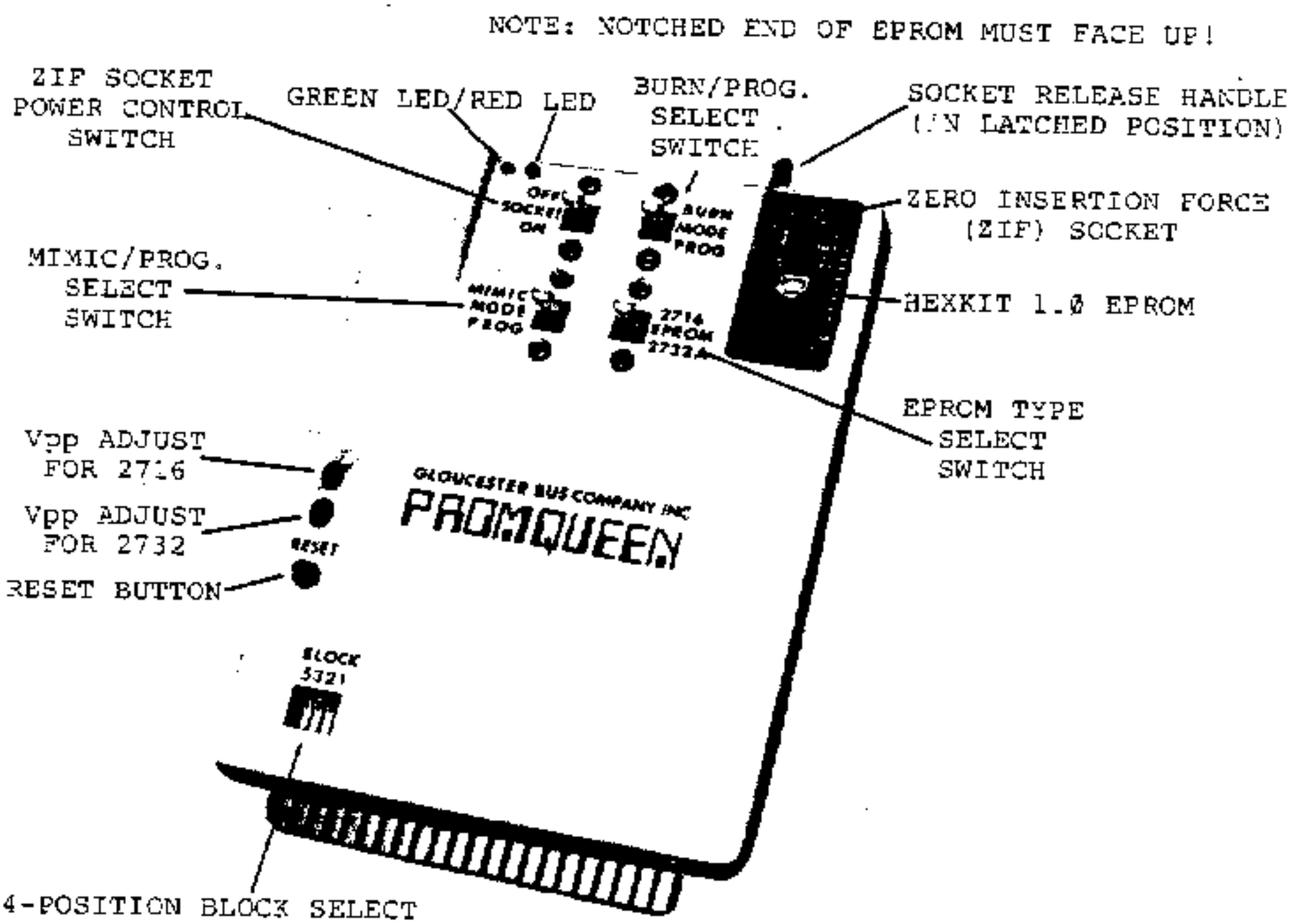
PROMQUEEN owners have a broad range of backgrounds, skills, and needs. Users that range from gifted teenage hobbyists to senior instrument development engineers are employing the PROMQUEEN to run music synthesizers, industrial processes, robotic devices, to develop not-so-standard classroom materials from basic math to college physics, and even to vend computer time.

With this in mind, the material in this manual has been arranged at varying levels of technical difficulty, and is geared to have you up and running as quickly as possible. It is assumed you have a working knowledge of BASIC, some background or a desire to build a background in assembly language, a good fundamental knowledge of computer architecture, numbering systems, and terminology, and a preference for creating devices, processes, games, whatever--your own way.

The manual is roughly divided into quarters. The first section is for the new user and contains applications you can use immediately while you are developing your skills. Then comes a full, sit-down tutorial of all the Hexkit software capabilities. Next, for the familiar user, comes a summary of all commands, routines, and procedures covered in the first section. These are written to help you find things fast. In the last sections, you'll find the technical hardware descriptions and advanced information and applications. PLEASE READ THIS MANUAL THOROUGHLY!

Our user hot line is (617) 283-7719.

PROMQUEEN CONTROLS



1.1 Plugging in Your PROMQUEEN CARTRIDGE the First Time

These steps along with the opposite diagram will quickly acquaint you with the PROMQUEEN controls.

1) Carefully examine the cartridge. If it appears damaged, return it for exchange. See page 79.

2) Remove the Hexkit program EPROM from the Zero Insertion Force socket. It will lift out easily after you have pulled up on the socket release handle. Set it on conductive foam.

3) Make sure the BLOCK select DIP switch is set for BLOCK 3, that is, the second rocker switch from the left is depressed at the top. Refer to the switch numbers embossed on the case, not the ones on the switches themselves. Use a dull pointed object to set BLOCK 3 and reset the other 3 blocks. The PROMQUEEN cartridge is now set to occupy the range of addresses from decimal 24576 ($6000) to decimal 32767 ($7FFF). Note that you can put the PROMQUEEN cartridge into any of the four expansion BLOCKS depending on which BLOCK SELECT DIP switch you set. Always avoid setting more than one BLOCK at a time. If more than one of the rockers is set on at one time, the VIC will not properly decode addresses.

4) Set the four toggle switches at the top of the cartridge so that they are flipped towards you, that is, with the toggle handles perpendicular to the top of the case. These settings will be:

    SOCKET=ON
    BURN/PROG=PROG
    MIMIC/PROG=PROG
    EPROM=2732A

5) Make sure the VIC is turned off. Plug the PROMQUEEN cartridge face up into VIC's expansion slot, which is the largest slot on the back of the VIC and the only port into which the PROMQUEEN will fit. Seat the PROMQUEEN firmly into the connector.

6) Turn the VIC on. Within 30 seconds the normal start-up message should appear on the monitor's screen. If it does not, remove the cartridge and check to make sure that only the BLOCK 3 rocker on the DIP switch is set, with all other rockers unset. If the normal start-up message does not appear with the rockers properly set, the cartridge is defective. Return it for exchange.

7) After the start-up message appears look to see that the green LED glows to indicate that the zero insertion force (ZIF) EPROM socket is connected through to the VIC and is receiving power. Flip the SOCKET switch to the OFF position. Observe that the green light goes out. Now it would be safe to install an EPROM in the socket, or remove an EPROM already installed without turning off the VIC. Don't install any EPROMs now. Flip the SOCKET switch back to the ON position. Leave the socket empty.

8) The MIMIC/PROG switch located below the SOCKET switch has these two functions: PROG enables the VIC to address the 4-kilobyte PROMQUEEN RAM, and MIMIC enables you to jumper to an external computer. If you flip the switch to MIMIC, the green LED goes out. Now, a jumper cable connected between the zero insertion force socket and an EPROM socket in an external computer, will enable the external computer to read (not write to) the contents of the PROMQUEEN RAM. Operating in MIMIC mode does not alter the contents of the PROMQUEEN RAM. The PROMQUEEN RAM appears to the external computer as though it occupies the same range of addresses as the external computer's EPROM socket. The VIC cannot address the PROMQUEEN RAM when the cartridge is in MIMIC mode.

BE CAREFUL NEVER TO ALLOW THE DC/DC CONVERTER TO COME ON (i.e., BURN/PROG=BURN AND MIMIC/PROG=PROG) WITH A MIMIC CABLE INSTALLED. IT COULD DAMAGE YOUR EXTERNAL COMPUTER. See page 59 of this manual for MIMIC mode instructions. For now, set the MIMIC/PROG switch back to PROG, and observe that the green LED comes back on.

9) The toggle to the right of the MIMIC/PROG switch is the EPROM select switch. This switch may be set to read and program either 2716 or 2732X EPROMs. A third type of EPROM, the 2732A, can be read when the switch is in the 2732 position, but it cannot be programmed until you adjust the voltage. See page 53. With the switch set for 2732 EPROMs, all 4096 bytes of the PROMQUEEN RAM can be addressed either by the VIC, when the MIMIC/PROG switch is set to PROG, or the external computer when the MIMIC/PROG switch is set to MIMIC. NOTE: You can also use 27C16 and 27C32 EPROMS with the PROMQUEEN (these are CMOS types). Flip the switch to the 2716 position. This does not affect the contents of the PROMQUEEN RAM, but does restrict access to only the first 2048 bytes of PROMQUEEN RAM by the external computer. No indicator lights are affected. This switch governs the built-in 21 or 25-volt programming voltage power supply, and the pin on the EPROM to which the supply is connected. ALWAYS MAKE SURE THAT YOU HAVE THIS SWITCH SET FOR THE PROPER EPROM TYPE BEFORE YOU ATTEMPT TO BURN AN EPROM. OTHERWISE YOU WILL PROBABLY DAMAGE THE EPROM WHEN YOU ATTEMPT TO BURN IT. (No damage will occur if you have the switch mis-set in the normal PROG mode). Flip the EPROM switch back to the 2732 position.

10) The upper right toggle has two modes: BURN and PROG . When set to BURN, this switch enables you to BURN your own program onto an EPROM. The RED LED will come on, indicating that programming voltage is being supplied to the EPROM socket. With no EPROM in the socket, test the BURN switch by slipping it away from you. Note that in BURN mode you can permanently program EPROMs either from BASIC (see page 48) or from Hexkit's BURN routine (see pages 15 or 44).

A faint buzzing noise made by the DC/DC converter inside the cartridge may be audible when the red LED is on. This is normal. Notice that the red LED will go out, along with the green if you now

flip the SOCKET switch to off, or if you flip the PROG/MIMIC switch to MIMIC. This is intended to inhibit the programming power supply in these modes. NEVER TURN OFF THE PROGRAMMING VOLTAGE THIS WAY WHEN YOU ARE BURNING AN EPROM. THE EPROM COULD BE DAMAGED.

You will not be able to read the contents of the PROMQUEEN RAM through the VIC when the PROG/BURN switch is set to BURN.

If the RED LED has been on all along, or if it fails to go out if you flip the PROG/BURN switch towards you into PROG mode, the cartridge is defective; return it for exchange.

Flip the PROG/BURN switch towards you into PROG mode. Observe that the red LED goes out, and the green LED stays on.

1.2 LOADING AND OPERATING HEXKIT

To install your Hexkit EPROM, first restore the toggle switches to these positions:

SOCKET=OFF
BURN/PROG=PROG
MIMIC/PROG=PROG
EPROM=2732

a) Unlatch the ZIF (EPROM) socket by pulling up on the handle with the greenish knob.

b) Put your Hexkit EPROM, notched end away from you, in the ZIF socket.

****

REMEMBER

Turn OFF the SOCKET switch whenever you change EPROMs.

****

c) Latch down the ZIF handle and turn on the SOCKET

switch. The green LED should come on.

d) To load Hexkit 1.0 when the cartridge occupies BLOCK 3, type:

SYS29039 <RETURN>

The Hexkit program will start to run by itself. If it doesn't, turn off the VIC and check the BLOCK DIP switch. See page 1, step 3.

e) You should now be seeing the Main Menu display:

```
EDIT HEX.........A
BURN EPROM.......B
SAVE ON DEVICE...C
LOAD FROM DEVICE.D
V'FY SAVE........E
```

f) For now, type an A to enter the EDIT HEX routine.

g) <u>Review:</u>

1. In BLOCK 3 type **SYS29039 <RETURN>** to load Hexkit and display the Main Menu.

2. Select the operation you want by typing A, B, C, D, or E. (No <RETURN> is needed.)

3. To get out of an operation and back to the Main Menu, type X or series of Xs.

4. To restore VIC BASIC, type an X to the Main Menu. If you go into VIC BASIC from the Main Menu, the Menu will not erase, but you will see the "ready" message and the blinking cursor.

5. To restore Hexkit (after it has been loaded by the SYS command) from VIC BASIC, type a RUN <RETURN>.

6. For now, if you wipe out entirely, go back to step 1, and reload Hexkit.

h) After you have typed an A to EDIT HEX on the Main Menu, you will see this display:

```
EDIT HEX
WHERE? -
```

i) Type $6000 in Hexadecimal and return.

```
EDIT HEX
WHERE? $6000
```

****

REMEMBER

The $ sign tells the computer you are entering a Hexadecimal number.

****

j) You are also able to enter decimal numbered addresses (preceded by #) or a MARKER. These are explained in the Hexkit Tutorial (see page 30).

k) You should see this display. (NOTE: Memory contents may be anything because the memory chips turn on with random data.)

EDIT HEX

24569	5FF9	5F
24570	5FFA	5F
24571	5FFB	5F
24572	5FFC	5F
24573	5FFD	5F
24574	5FFE	5F
24575	5FFF	5F
24576	**6000**	**00**
24577	6001	A2
24578	6002	00
24579	6003	FF
24580	6004	00
24581	6005	FF
24582	6006	00
24583	6007	FF

l) You are now ready to communicate directly with the first address (as set in BLOCK 3) of the PROMQUEEN user RAM.
Look at the line that the blinking cursor is on:

```
24576 = Address in decimal
 6000 = Address in hexadecimal
   00 = One byte (8 bits), the present
        contents of address $6000 in Hex
```

You will be using an additional leftmost column and a rightmost column (pages 29 and 30). These will be explained in the Hexkit Tutorial.

m) Turn the SOCKET switch OFF, release the ZIF socket handle, and remove the Hexkit EPROM. Notice that Hexkit is still running, because you have loaded it into the VIC. You could now install another EPROM to copy it. (See page 15 for the copy procedure.)

For now try the following:

n) Type the little program in hexadecimal code exactly as given below, on the VIC keyboard. Observe what happens to the display as you do this. You don't need $ signs to enter hex code.

```
A2 93 8A 20 D2 FF CA E0 21 30 F7 60
```

If you make any mistakes, just finish the entry by typing any hex digit and back up the display by using the cursor up key. Type over to change. Move the display up or down by using the cursor keys. Check your list of hex numbers which must start with A2 at 24576 ($6000).

o) The Hexkit Tutorial gives you an INITIATE keystroke command, but for now, you can run your program through BASIC. (This way, you'll know two methods of initiating your machine code programs.) Type X (for the Main Menu), X (to get back to BASIC), SYS24576 <RETURN> (to access the decimal address of your program), and RUN <RETURN>.

p) You should now be seeing the character set of the VIC displayed. If, instead, something went wrong, or the cursor is lost, you can RESET. Otherwise, just type RUN <RETURN> to get back into Hexkit.

q) Press the RESET button on the PQ cartridge. This gives you the VIC's cold start display. You can reload Hexkit as in review step 1 on page 5, or you can just restart Hexkit by typing SYS 4112. You will be happy to know that your machine code programs will remain intact after a RESET. (The actual text of BASIC programs will also be unaffected BUT the "pointers" that tell the operating system where the BASIC is will be reset. Hence, BASIC programs will not run after resets.) You'll notice that whatever you entered into PROMQUEEN RAM will still be there. Using the RESET to recover from faulty machine code won't alter the code. This saves you the tedium of storing machine language programs before testing them.

****

REMEMBER

You can always get back to the Main Menu by typing an X or a series of Xs depending on where you are in the program.

Typing an X to the Main Menu will restore you to VIC BASIC.

Pressing the RESET buttom gives you a clean start. You can get back up from EPROM by typing SYS 29039 (if you are in BLOCK 3) or SYS 4112 if Hexkit is already loaded into VIC RAM.

****

PROMQUEEN AS VIC 4K EXPANSION

Here is an easy way to expand your VIC. Turn off the VIC and remove the cartridge. Turn off the BLOCK 3 DIP switch, and set ON the BLOCK 1 DIP switch. Plug in the cartridge and turn on the VIC. Notice that the "bytes free" message now says 7679. You have added the PROMQUEEN RAM to the VIC BASIC RAM, and relocated the position of screen memory in the VIC address space. You

can now write BASIC programs more than twice as long as before, or use the PROMQUEEN in BLOCK 1 (address space of 8192 ($2000) to 16383 ($3FFF) for machine code programs and EPROMs. Note that you can put the cartridge into any of the 4 VIC expansion blocks just by setting the appropriate switch on the BLOCK select DIP switch.

Hexkit is designed to be loadable to the VIC's memory from any of the 4 expansion blocks into which you can place the cartridge. The command to load Hexkit when the cartridge is in Block 1 is SYS 12619. Try it and see. The funny business you see flash briefly on the screen occurs because the load commands always put the screen memory back to its usual location in the unexpanded VIC. PROMQUEEN addressing is fully explained in the Hexkit Tutorial.

SUMMARY OF HEXKIT LOAD COMMANDS:

BLOCK	COMMAND
1	SYS 12619
2	SYS 20829
3	SYS 29039
4	SYS 45441

1.3 SAVING MACHINE CODE PROGRAMS

The purpose of this section is to enable you to make your backup copies of Hexkit while learning how to use these routines:

- O Save, Load and Verify to Device
- O Burn EPROM

1.3.1 To SAVE Hexkit (or your own machine code program) on TAPE, follow these steps:

a) In VIC immediate mode, use the usual VIC VERIFY to wind the cassette to where you want the tape to be.

b) Turn OFF the VIC, remove the PROMQUEEN, and reset the DIP Switch to BLOCK 3. (You'll know how to save from other blocks after you read the Hexkit Tutorial.)

c) With the Hexkit EPROM installed (notch up) in the ZIF socket, turn on the VIC and the SOCKET switch, type SYS29039 <RETURN>.

d) Select Menu option C. The display:

```
SAVE ON DEV
DEV.# -
```

e) Type a 1. The display:

```
SAVE ON DEV
DEV. # 1
FROM -
```

f) Type $1001 <RETURN> (starting address of Hexkit program code in Hex)

```
FROM $1001 TO -
```

g) Type $1DEE <RETURN> (ending address in Hex)

```
FROM $1001 TO $1DEE
```

In this case, you are saving the Hexkit program directly from the above range of addresses in VIC user RAM. Note that when saving any block of code, give the starting address of the code to the FROM? prompt and the ending address to the TO prompt.

h) To the NAME? prompt, type Hexkit or whatever (up to 16-letter) name you wish and <RETURN>.

i) Follow the instructions to "Press record and play on tape." When the SAVE is complete, the Main Menu returns to the screen. (Note that you cannot put an end-of-tape marker on tapes through Hexkit.) The code will be saved under the name you gave it with a reload header affixed to the file.

More technical information and suggestions for use of the additional capabilities of this routine can be found on page 49. Although you cannot save programs to tape or disk from BLOCK 5, in any other block you can save directly off the EPROM or from the PROMQUEEN user RAM.

j) If you have a problem, you will see a VIC Kernal error message instead of the ones with which you might be more familiar. Refer to the "VIC-20 Programmer's Reference Manual" page 210 for the list of IO error codes.

2. TO VERIFY Hexkit (or your machine code program) after the save on tape, follow these steps:

a) Don't forget to rewind your tape.

b) Type an E to the Main Menu. The display:

```
V'FY SAVE
DEV.# -
```

c) Type a 1. The display:

```
V'FY SAVE
DEV.#1
NAME? -
```

d) Type the name you have used to the NAME? prompt and <RETURN>.

e) After you "press play on tape," if the VERIFY is good, you'll see an OK message and the cursor. Type any key to get back to the Main Menu.

f) If the VERIFY is not good, you will receive a VIC Kernal IO ERROR message, or the display will just say ERROR. If this happens, first check to make sure you typed the device number and the name of your program correctly. You can get back to the Main Menu by typing any key. Try again.

1.3.2 To SAVE Hexkit (or your machine code program) on DISK, follow these steps:

a) The PROMQUEEN should still be set to run in BLOCK 3. If it isn't, reset the DIP switch for BLOCK 3. (You have to turn the VIC off and remove the cartridge to reset this switch.)

b) Plug in the PROMQUEEN, and if the Hexkit EPROM is not in the ZIF socket, turn OFF the SOCKET switch before dropping Hexkit (notched end up) back into the ZIF socket.

c) After initializing the disk, turn on the SOCKET switch and load Hexkit into VIC user RAM by typing SYS29039 <RETURN>.

d) Select Menu option C. The display:

```
SAVE ON DEV
DEV.#
```

e) The disk device number (DEV#) is 8. Type an 8. The display:

```
SAVE ON DEV
DEV.#8
FROM _
```

f) Type in $1001 <RETURN>. (This is the starting address of the Hexkit program in VIC RAM.) The display:

```
SAVE ON DEV
DEV.#8
FROM $1001 TO -
```

g) Type in $1DEE <RETURN>. (Ending address of Hexkit.) The display:

```
SAVE ON DEV
DEV.#8
FROM $1001 TO $1DEE
```

NOTE: If you make a mistake typing the addresses, type X a couple of times, press <RETURN>, and you will be returned to the preceding prompt.

h) Type HEXKIT to the NAME? prompt. The disk should begin to save, and "SAVING HEXKIT" will appear on the screen.

i) If the SAVE is good, you'll be returned to the Main Menu.

j) To VERIFY that Hexkit (or your machine code program) has been saved on DISK, type E to the Main Menu, 8 to DEV.#, Hexkit (or * which tells the operating system to VERIFY the last program saved on disk) to NAME? and the verifying process will begin. The display:

```
V'FY SAVE
DEV.#8
NAME? Hexkit
Searching for Hexkit
Verifying
OK
```

k) OK - means that the program has been saved and verified. ERROR - means that something went wrong. Type any key to return to the Main Menu.

l) You can reload with the VIC's normal load command. Hit RESET, type LOAD"HEXKIT",8 and after the load, type SYS4112, or alternatively, type RUN <RETURN>. This works for tape as well, but does not work on 8K and expanded VICs.

1.3.3 To LOAD from TAPE or DISK using Hexkit, type D to the Main Menu when it is displayed.

a) You will be asked if you want to load your program to an area other than from where it was saved. To the prompt NEW LOC? you can answer Y or N.

An N will load the program back into the VIC exactly where it was saved from. Be sure and have RAM there to receive it. You might get an "out of memory e rror." Hit RUN/STOP (RESTORE) if this happens.

A Y will generate this display:

```
LOAD PRM DEV
DEV.# 1 (or 8 for disk)
NEW LOC? Y
WHERE?
```

b) Specify the new starting address and your program will be loaded starting there. You will see the normal prompts and Kernal error messages. You can LOAD Hexkit or another machine code program into the PROMQUEEN RAM in BLOCK 3 (instead of the VIC RAM) by typing $6000 to the WHERE?

prompt (provided you said "Y" to the "NEWLOC" prompt). This is called a relocate load. This (and other advanced SAVE and LOAD information) can be found starting on page 49.

VIC TIP: If you load "PROGNAME", 1, 3 (giving a secondary address of 3 to the VIC's normal load (command), the program will load to the range from where it was saved, not where the VIC wants to put it.

1.3.4 To BURN your backup Hexkit EPROM, follow these steps which can also be used for copying any other compatible EPROM.

NOTE: The addresses used for this procedure apply only to BLOCK 3, and these instructions are written only to acquaint you very quickly with PROMQUEEN operations. The extensive options you have for relocating or remapping code are discussed on pages 32 to 35. If using VICMON, see pages 59 and 60.

a) Check that the DIP switch is set to BLOCK 3, set the toggle switches for SOCKET=OFF PROG/MIMIC=PROG PROG/BURN=PROG and EPROM=2732. Install the Hexkit EPROM, notched end up.

b) Plug in the PQ and turn on the VIC. Turn on the SOCKET and load Hexkit with SYS29039 <RETURN>.

c) Type an A to the Main Menu, and type $6000 <RETURN> to the WHERE? prompt.

d) The 15-line display as on page 6 should appear. Type a T. This is one of the 22 keystroke edit commands you will learn about in the Hexkit Tutorial. This one is the TRANSFER CODE command. You will be transferring the code from the Hexkit EPROM to the PROMQUEEN cartridge RAM. Code has to be in the PROMQUEEN RAM before it can be

burned onto an EPROM.

e) A prompt window will appear at the bottom of the display:

```
-----------------------------------
        XFER FRM -
```

Type $7000 <RETURN> to the XFER FRM prompt

f) Then below what you just typed the prompt STOP@ 0000 will appear, cursor blinking on the first 0. Type over the 0's with 6FFF and hit <RETURN>.

```
-----------------------------------
        XFER FRM $7000
        STOP@ 6FFF
```

XFER FRM defines the first address in the EPROM (as it is seen by the VIC when the PROMQUEEN is set to operate in BLOCK 3). STOP @ defines the LAST address you want to use in the workspace of the PROMQUEEN RAM. You can think of STOP@ as something like a line on a measuring cup. You are "pouring" code from the EPROM, starting from $7000 into the PROMQUEEN RAM and up to STOP@. The $6000 you called for in step c is the "start at" or bottom of the measuring cup, or first address of PROMQUEEN RAM in BLOCK 3.

g) Hexkit has now been copied from the VIC RAM into the PQ RAM and is ready to be copied (BURNED) on an EPROM.

h) Turn OFF the SOCKET switch, remove the Hexkit EPROM, put in a blank 2732 EPROM, and turn the SOCKET switch ON.

i) Type X to get back to the Main Menu and B to access the BURN routine. The display:

```
BURN EPROM
FROM
```

j) Type $7000 <RETURN> (lowest address in PQ RAM + 1000) to the FROM prompt and $7FFF <RETURN> to the TO prompt.

k) If the screen turns black and says

```
ERASE EPROM
```

the program found data other than $FF in the address range you gave. Type an X to respecify addresses, or try another EPROM (don't forget to turn off the SOCKET during the change). The program will retest for erasure when you type a <RETURN>.

l) If all $FF's were found in the given range of addresses, the program will instruct you to set the BURN/PROG switch to BURN. Make sure the EPROM type setting is correct before doing so. When you flip the switch to BURN, the red LED on the PROMQUEEN comes on. It tells you the program voltage power supply in the PROMQUEEN is working.

m) Hit a <RETURN> to start the BURN process. No actual burning occurs (unless the EPROM is in backwards) until you tell it to start with this <RETURN>.

****

REMEMBER

The BURN routine will prompt you to set switches, etc. After setting the switch, hit <RETURN> to continue.

****

n) The screen will now be red, and a counter will be running to show the addresses as they are being burned. It takes 70 milliseconds to BURN each byte of data, a whole 4K 2732 takes just under 5 minutes.

o) When the BURN is complete, the screen will turn blue and the display:

SET SWITCH TO PROG -

appears

p) Set BURN/PROG back to PROG and hit <RETURN>. The display:

BURN VERIFIED _

Your EPROM copy of Hexkit has been BURNED and VERIFIED. Type <RETURN> for the Main Menu.

q) If instead you see ERROR at some hex address, the PROMQUEEN is telling you that the data found in this address on the EPROM is not identical to the data in the corresponding address ($1000 bytes lower in memory) in the PROMQUEEN RAM. The program will let you continue the verification process by typing more <RETURN>'s, or let you go back to the Main Menu by typing an X. If the BURN has errors, recheck your procedure (is the PROMQUEEN in the right block?, etc.) If the procedure and the programming voltage (see page 51) are good, try another EPROM.

r) BURN VERIFIED means the BURN was good. Turn the VIC off, then back on again. See that the program loads normally with the command SYS29039 <RETURN>.

1.4 SUMMARY: EPROM DUPLICATION

If you're in BLOCK 3, you can copy any 2732 EPROM by using the procedure on pages 16-18 and summarized below:

a) Load Hexkit into VIC user RAM (see the table on page 9). This is SYS 29039 in BLOCK 3.

b) Turn OFF the SOCKET switch, replace Hexkit with the EPROM to be copied, turn ON the SOCKET, and transfer the contents of the EPROM to the PQ RAM by these steps:

   1) Type A (to the Main Menu), give the starting address of the PROMQUEEN RAM (which is $6000 in BLOCK 3) then type a T for the transfer function (you transfer from $7000 in BLOCK 3 and give the STOP@ as 6FFF).

   2) Then follow steps 1) through 15) in the BURN routine summary on page 58.

c) If you don't want to copy the entire EPROM, just add $1000 to the lowest address and $1000 to the highest address of the contents of the PQ RAM that you do want to BURN. Give those addresses to the prompts of the BURN routine.

This completes a brief walking tour of the PROMQUEEN cartridge. To learn more of the many things you can do with it, read on.

1.5 HOW TO SAVE BASIC ON EPROMS WITH THE PROMQUEEN

Try the following experiment:

1) Set your PQ to work in BLOCK 5. Remove any RAM expansion on your VIC, turn off the VIC, and plug in the PQ. Put your Hexkit 1.0 EPROM (notched end up) in the socket and turn the SOCKET on. Turn on the VIC.

2) Load a BASIC program from cassette or disk. Be sure the program fits in an unexpanded VIC.

3) Briefly RUN the program; then, stop it with the RUN STOP key. Clear the screen.

4) Type SYS45312 <RETURN>. After the cursor disappears momentarily, it will return with a "Ready" message.

5) Type NEW <RETURN> to obliterate your BASIC program.

6) Tap the RESET button on the PQ cartridge.

You should now be seeing your BASIC program running! If you try to stop it with the STOP key, you will find that you cannot! If you hit the RESET button again, it will only start the BASIC program over. You see that you have locked out the user just as though your BASIC program were a game cartridge. You'll have to turn off the VIC to clear this condition.

This SYS45312 routine provided on your Hexkit EPROM takes care of everything needed to save BASIC programs from a VIC (even an expanded one) onto a single EPROM chip, complete with "AUTO START." The procedure is very simple. For any 3.5K BASIC program you want to save on EPROM, just:

1) Set the PROMQUEEN to run in BLOCK 5 and plug it back into the VIC. Put the Hexkit EPROM in the EPROM socket and turn the socket on. Turn on the VIC, but don't load Hexkit.

2) Load the BASIC program from tape, disk, modem, however you want to do it.

3) Run the BASIC program to make sure the pointers are properly set. (See the VIC-20 "Programmer's Reference Guide" for the VIC's memory map. You'll find that BASIC keeps track of where it starts, where the variables are, etc. in locations $2B to $38. These are the pointers in question. Stop the program. Type SYS45312 <RETURN> to save the BASIC into the PQ RAM ready to burn onto the EPROM.

4) Load Hexkit off the EPROM with SYS45441. Select Menu option B to burn the EPROM. Put a clear EPROM in the PQ and BURN from addresses $B000 to $BFFF (or less, if your BASIC is shorter). If your BASIC is really brief, it may fit on a 2716, otherwise, use a 2732 or 2732A. Once the BURN procedure is complete and VERIFIED, X yourself back to VIC BASIC, and, since you're in BLOCK 5, type SYS45441. Your BASIC program should begin running.
That's all there is to it.

The EPROMs made with the aid of SYS 45312 have a header put into the first 512 bytes that gives these EPROMs unusual properties.

1) You will be able to reload and run the BASIC program off the EPROM, from whatever possible location a 4K EPROM socket can appear, anywhere in the VIC's expansion space. All you have to do is use the right SYS command to load and run the BASIC off the EPROM. You do NOT need a "run time package" to make this possible. Various VIC-20 expansion cards give various possibilities of "where" to plug in the EPROM.

2) If you plug an EPROM made with the aid of SYS 45312 into a circuit card that has a socket with a start address of $A000, the VIC will be auto-started with the BASIC program, and the STOP key will be disabled.

There are eight possible locations in the VIC's expansion space where a 4K EPROM can be put. The possibilities, along with the SYS commands the LOAD and RUN the EPROMed BASIC commands are listed below:

VIC EXP BLOCK	EPROM SOCKET ADDRESS HEX	EPROM SOCKET START ADDRESS DEC	COMMAND TO LOAD AND RUN BASIC FROM EPROM TO UNEXPANDED VIC (1)	COMMAND TO LOAD AND RUN BASIC FROM EPROM TO EXPANDED VIC (2)
1	$2000	8192	SYS 8514	XXXXXXXX
	$3000	12288	SYS 12619	SYS12682
2	$4000	16384	SYS 16724	SYS16784
	$5000	20480	SYS 20829	SYS20886
3	$6000	24576	SYS 24934	SYS24988
	$7000	28672	SYS 29039	SYS29090
5	$A000	40960	SYS 41336	SYS41384
	$B000	45056	SYS 45441	SYS45486

(1) The commands all put the screen at $1E00 first.

(2) These commands do not relocate the screen before loading the BASIC.

If you have an expansion card, and it takes, or you doctor it to take, 4K EPROMs, you will find that the address location of the EPROM socket will be one of the address ranges from the table above.

The EPROM location at $A000 is special. At this location, an EPROM made with SYS 45312 will, on power up, autoload and start the BASIC program you've burned on it, provided that it's for an unexpanded VIC. If you want to have AUTO START for an expanded VIC, see pages 68 to 72. Put the EPROM anywhere else, and you can, from the keyboard or from software, use the relevant SYS command from the table above to load and RUN the EPROMed BASIC.

If you are putting several BASIC programs on the up-to-eight 4K EPROMs you can put in the VIC's expansion space, you will want to reserve this auto starting $A000 location for a main menu you can write in BASIC and burn on EPROM with the aid of SYS 45312.

Remember also that an auto start or reset through an EPROM made with SYS 45312 and located at $A000 will lock out the STOP key. You probably want this feature in your finished program, but you must remember to include in your BASIC program some way to stop it through software during development. The STOP key can be reenabled by executing a POKE 808,112 command. You might save time if you put this command at the start of your program, and delete it only when ready to BURN the final version.

2.1 THE HEXKIT 1.0 TUTORIAL INTRODUCTION

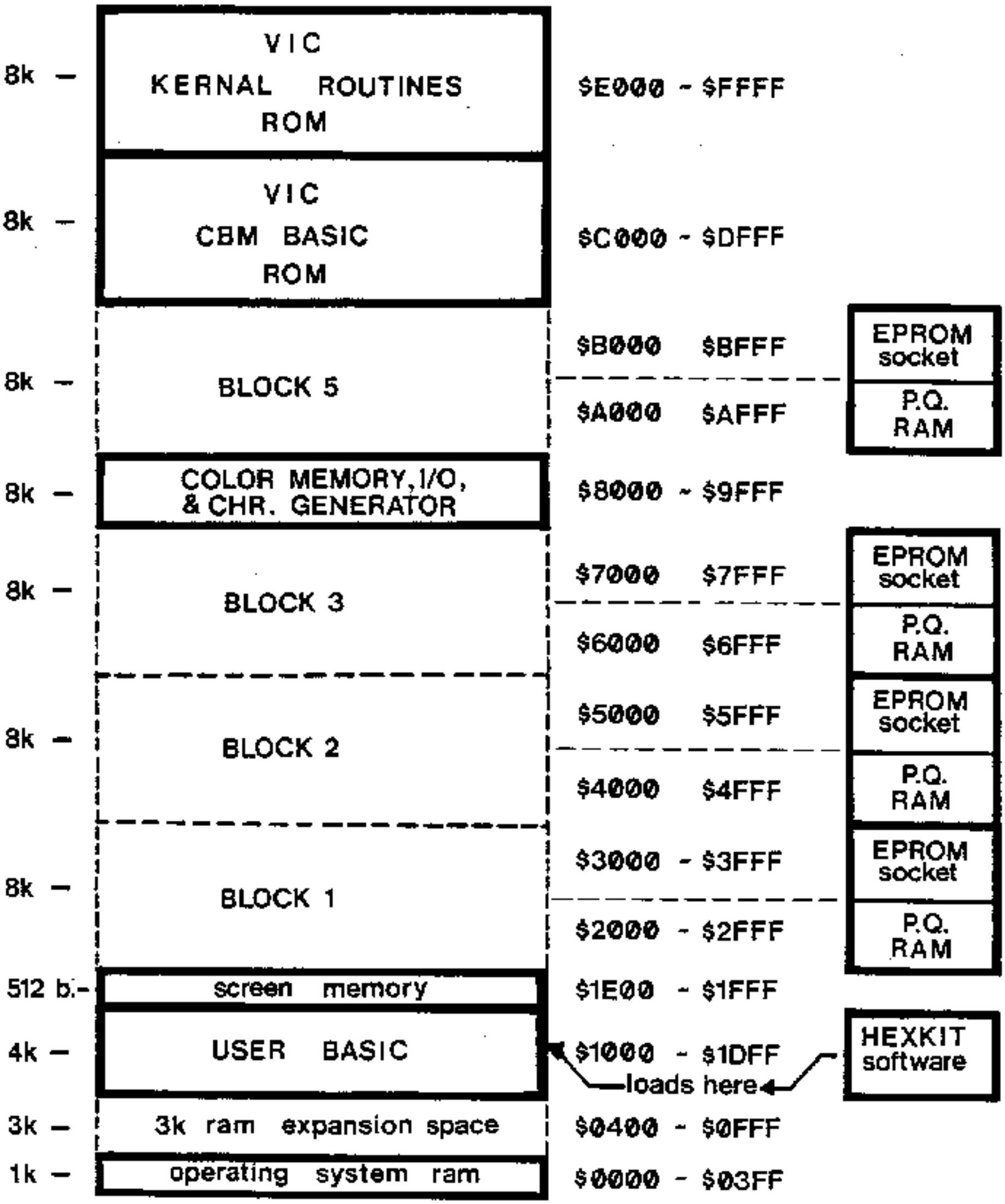
Hexkit 1.0 provides the user with a set of utilities for entering, editing, and debugging other programs written in machine code. Hexkit also gives a handy set of utilities for text string generation, and for using tape and disk for program storage.

The program is designed to load into the standard user memory of the Commodore VIC-20 at address $1001 to $1DFF. It is written in 6502 machine code. It leaves four kilobytes of RAM in the PROMQUEEN cartridge open for your program development, and four kilobytes are available for your own EPROMs when you drop them into the PROMQUEEN ZIP socket. You select the locations of these two areas in the address space of the VIC when you set the BLOCK SELECT DIP switch on the cartridge before you plug it in. The program is written to load to VIC BASIC memory so that you won't need an expanded VIC to use the PROMQUEEN as a stand-alone development system.

You can use the full set of Hexkit utilities when writing code for 6502, Z-80, 8080, 8085, NSC-800, and any other 8-bit microprocessor which, like the 6502, requires that 16-bit addresses be given with the lower-order byte first. Some of the utilities will not work with 6800, 6809, 1802, and other microprocessors which expect the high-order byte of an absolute address to be given first. If you are using Hexkit to write code for any microprocessor other than the 6502, you will not be able to test your program on the VIC's 6502, but you can, of course, use the PROMQUEEN's MIMIC mode to test your program directly on the target processor. Regardless of the processor for which your program is written, you can use 2716, 2732, and 2732A EPROMs as well as the VIC's disk drive and cassette recorder for your program storage.

Hexkit 1.0 requires 3583 bytes of RAM. When loaded, the program starts at address $1001 and ends at address $1DEE (4097 to 8191 in decimal) in the VIC.

MEMORY MAP


8k
VIC
KERNAL ROUTINES
ROM
$E000 - $FFFF
8k
VIC
CBM BASIC
ROM
$C000 - $DFFF
8k
BLOCK 5
$B000 $BFFF
$A000 $AFFF
EPROM socket
P.Q. RAM
8k
COLOR MEMORY, I/O,
& CHR. GENERATOR
$8000 - $9FFF
8k
BLOCK 3
$7000 $7FFF
$6000 $6FFF
EPROM socket
P.Q. RAM
8k
BLOCK 2
$5000 $5FFF
$4000 $4FFF
EPROM socket
P.Q. RAM
8k
BLOCK 1
$3000 - $3FFF
$2000 - $2FFF
EPROM socket
P.Q. RAM
512 b.
screen memory
$1E00 - $1FFF
4k
USER BASIC
$1000 - $1DFF
loads here
HEXKIT software
3k
3k ram expansion space
$0400 - $0FFF
1k
operating system ram
$0000 - $03FF

2.2 SYS COMMANDS TO LOAD HEXKIT

Hexkit is loaded from its EPROM into the VIC by a SYS command executed in immediate mode. The address to use with the SYS command depends on the expansion block in which the PROMQUEEN is set. The starting address of the space available for user programs also depends on the BLOCK switch setting:

SYS TO LOAD		USER PROGRAM DEVELOPMENT SPACE
BLOCK 1	SYS 12619	$2000 - $2FFF
BLOCK 2	SYS 20829	$4000 - $4FFF
BLOCK 3	SYS 29039	$6000 - $6FFF
BLOCK 5	SYS 45441	$A000 - $AFFF

The screen will be put back at $1E00 when you load Hexkit.

2.3 PROMQUEEN, HEXKIT, AND VIC ADDRESSING

Study the simplified memory map diagram on the opposite page. Notice which memory allocations belong to the VIC and which are available to the PROMQUEEN.

There are four 8K expansion blocks made available by the VIC. These are rather like vacant lots: the addresses are assigned, but the houses aren't there. Think of the PROMQUEEN user RAM and the Hexkit EPROM (ZIF) SOCKET as two "mobile homes" that you can move around. Their addresses depend on the DIP switch BLOCK setting. If, for example, you set the DIP switch to 1, the PROMQUEEN RAM will occupy "space" at addresses 12288-16383($2000-$2FFF), with the EPROM socket in addresses $3000-$3FFF, right next door and also in BLOCK 1.

Take another look at the diagram. What addresses would the PROMQUEEN RAM occupy if the DIP, or BLOCK, SWITCH were set at 5? What addresses would the PROMQUEEN EPROM SOCKET occupy if the DIP switch were set at 3?

When you type such commands as SYS 12619 (DIP=BLOCK 1) the VIC looks for address 12619, finds it on the Hexkit EPROM in the EPROM SOCKET and begins to execute the machine code program that it finds there. This program loads the Hexkit operating software into the VIC user BASIC RAM. (after resetting the screen to $1E00). Regardless which BLOCK you use, Hexkit always occupies 4097-7679 ($1001-$1DFF) in VIC BASIC RAM.

A later section in this manual tells you how to "remap" Hexkit to run where you want it to if you want to free up the space in VIC user RAM from $1000 to $1DFF.

You will be writing your programs in the PROMQUEEN RAM, and the addresses you use will depend on which BLOCK switch you use. For example, in BLOCK 2, the user program space is from address $4000-$4FFF. When you BURN your programs onto an EPROM, you will be referencing EPROM socket addresses, which in this case, would be $5000 to $5FFF. This lets you put your program anywhere you want to on the EPROM as long as the addresses correspond. For example, if you have a program in the PROMQUEEN RAM that runs from addresses $4009 to $4BBB, you'll be burning it at $5009 to $5BBB.

Look at the diagram. If you have a program at $6ABC to $6DDD, what BLOCK are you in? Are you in PROMQUEEN RAM or in the SOCKET? If you want to BURN this program, what addresses do you use?

Study the memory map in the VIC programmer's reference and the tables on pages 25 and 54. You will see that you have four choices where you can set the cartridge. To work all the practice examples given in the next few pages, the cartridge must be set to operate in BLOCK 3. This means the PROMQUEEN RAM can be used for our program development in the range of addresses from $6000 to $6FFF, and any EPROMs in the PQ's ZIF socket can be accessed by adding $1000 to the current PROMQUEEN RAM space addresses. (In BLOCK 3, EPROM access = $7000 to $7FFF.) A code routine on the HEXKIT EPROM will download the whole Hexkit program into the VIC's memory when you type SYS29039 <RETURN>.

2.4 LOADING HEXKIT FOR A PRACTICE RUN

Make sure your VIC is turned off. Remove the PROMQUEEN cartridge from the VIC and set it to run in BLOCK 3. The toggle for BLOCK 3 is the second from the left on the BLOCK DIP switch. Set the top of the rocker switch in. Remove any other expansion you may have on your VIC that occupies BLOCK 3. Turn up the sound, if any, on your monitor.

Plug the cartridge back into the VIC's expansion port, turn on the VIC and turn OFF the SOCKET switch. If you will be using the disk drive, you must insert your disk and initialize it before you start Hexkit 1.0.* The program does not do this for you. Execute this BASIC statement in immediate mode: OPEN1,8,15,"I" <RETURN>

Put your Hexkit EPROM in the ZIF socket. Remember that you must turn OFF the SOCKET switch whenever you change EPROMs. The notched end of the EPROM must face the top of the cartridge. Latch it down and turn SOCKET switch ON. The other three switches should be vertical. You should see the green LED on the cartridge come on. If the green LED isn't on, you can't read the contents of the EPROM.

To load Hexkit 1.0 when the cartridge occupies BLOCK 3 you type: SYS29039 <RETURN>

The program will start to run by itself.

The Main Menu is displaying the five main options provided by Hexkit:

```
EDIT HEX.............A
BURN EPROM...........B
SAVE ON DEVICE.......C
LOAD FROM DEV........D
V'FY SAVE............E
```

The option is selected by typing A, B, C, D, or E, and no <RETURN> is necessary. Once in any Menu option you can type an X to get back to the Main Menu. Typing an X from the Main Menu returns you to the VIC's BASIC operating system.

* NOTE: You must also close the disk channel after saving to disk, but before you turn off the drive, so as to update the directory.

You can stop the execution of Hexkit whenever the Main Menu is displayed by typing an X. If you do so, the Menu is not cleared from the screen, but you'll hear the sound effect, and see the "READY" message and cursor appear at the bottom of the screen. You are back in VIC BASIC.

To get back to Hexkit from the VIC BASIC operating system, clear the screen and type RUN <RETURN>.

The RESET button on the PROMQUEEN cartridge causes a "cold start" to be executed by the VIC. If you tap it you will get no result if you then type RUN, but you can still restart Hexkit by typing: SYS 4112 <RETURN>. This is because the cold start wiped out the single line of BASIC in Hexkit which is: 1 SYS 4112:END.

Stop the program by typing an X. Enter the above line of BASIC followed by a RETURN. Then, type RUN <RETURN>. This restores the RUN start.

If you enter any BASIC programs with numbered statements when Hexkit is loaded into the VIC BASIC memory, you will probably be typing over Hexkit, wiping out some of it. Just reload Hexkit when you are finished with the BASIC by using the appropriate SYS LOAD command from the table on page 25. You can execute any BASIC commands in immediate mode when Hexkit is loaded but not running, without causing it any damage. Hexkit does not protect itself from BASIC by resetting any pointers.

You can also "cold start" Hexkit by simply typing SYS 4112 <RETURN> or SYS (wherever your Hexkit is located) if you have remapped it elsewhere.

2.4.1 USING THE EDIT HEX . . . A

After you select the option to EDIT HEX by typing an A, the menu will disappear, and you will see the following:

```
EDIT HEX
WHERE?
```

2.4.2 USING #, $, OR MARKER TO SPECIFY ADDRESSES

There are three ways to tell the program where you want to Edit Hex. You can give the address in decimal if you precede it by a "#" sign and press RETURN after your entry. You can give the address in Hex if you precede it by a "$" sign and press RETURN. You can also type an M and a Hex number from 0 to F to reference a previously assigned "MARKER." You will be able to practice using $, #, or M in the section below. No need to press RETURN when referencing a MARKER. In addition, if you had selected EDIT HEX by mistake, you could pop back to the Main Menu by typing X.

To the WHERE? prompt of EDIT HEX type the address $6000 in Hex and then <RETURN>.

You should see the EDIT HEX display. If you look at the line that the cursor is blinking on you should see the decimal address #24576, Hex address $6000, and a byte of machine code. The cursor should be blinking on the high-order nybble of the code.

Press the X key, to get back to the Main Menu. Select EDIT HEX again by typing the an A. This time, give the same address in decimal by typing #24576 <RETURN>. You'll see the same display. (Nothing happens unless you type the #.)

Now, type an M. You'll see the prompt "MARK" appear in the prompts area at the bottom of the screen. At the same time, an M will appear at address $6000 in the leftmost column of the display, the cursor blinking next to it. Type a 0 (zero). You see that you have MARKED location $6000 as M0.

Type an X to get the menu, an A to get EDIT HEX, and this time answer the WHERE? prompt with M0. You'll get back to your marker at $6000 (24576) (M0).

Look at the display. Five columns of information are available. The leftmost column is for MARKERs. The next column is the address in decimal to make it convenient for interfacing your code programs to BASIC. The third column is the address in hexadecimal. The fourth column

has the current contents of the address given as a two-digit Hex number from 0 to FF (0 to 255 in decimal). The fifth column may not have anything displayed now, but it shows any VIC characters represented by the code in the fourth column. Notice that 15 addresses are displayed this way so that you can see the contents of the range of addresses around the address you have selected to edit.

2.4.3 SCROLLING

You can SCROLL the display in either direction with the CURSOR UP and CURSOR DOWN keys. On CURSOR DOWN, the display scrolls to lower-number addresses, CURSOR UP scrolls to higher-number addresses. The cursor keys are repeating and silent. If you want to be sure to scroll only one step at a time and hear a beep so that you can hear your progress while looking at your notes, use the f5 SPECIAL FUNCTION key to scroll to lower-number addresses, or the f7 SPECIAL FUNCTION key to scroll to higher-number addresses. Scroll around by various means to various addresses.

Whenever the display stops, you'll see the cursor blinking on the first digit of the hexadecimal contents of the middle address displayed. The blinking cursor shows where your editing entries will be stored. Scroll the display to some address above $6000 and stop. Now, try typing some numbers in hexadecimal. Hear the sounds and observe the display. You'll see the numbers you type entered in the cursor location, and the display will scroll to higher addresses. It takes two keystrokes to enter a byte of hexadecimal code.

TO GET BACK TO A TARGET ADDRESS when editing, use the f8 (shift/f7) SPECIAL FUNCTION key. For example, try scrolling some distance from $6000. Type f8. This will generate the DISPLAY prompt. Type $6000. The program will take you to any address you specify to the DISPLAY prompt of the f8 key. Note that the f8 DISPLAY prompt accepts $Hex, #Dec or Mn MARKER where n = the number you assigned the MARKER. The asterisk merely indicates where you left the program when you called for a special function. It is not a "Permanent marker."

2.4.4 **IF YOU MAKE A MISTAKE** on the first nybble of code, you can abort the entry by typing an X. Try it: type one Hex digit, then an X, and observe that the cursor backs up to the first digit (but the display does not scroll). If the cursor has been blinking on the second Hex digit of the edit address, you'll find that function keystrokes don't work: you'll hear the auditory feedback - raspberry - for an invalid keystroke. SPECIAL FUNCTION keys work only from the leftmost nybble of code. If you hear the raspberry (invalid entry) audio feedback, get back on the first digit by typing an X. If you type any SPECIAL FUNCTION KEY by mistake, you can abort by typing an X. If you press $, #, or M, you can still abort by typing an X after the M or by typing at least 2 X's after the $ or # <RETURN>. Note that you don't need a RETURN after a MARKER reference. Any invalid number entries also cause an abort. The program does not accept any commands except those discussed in this tutorial. If you type addresses larger than 4 or 5 digits (Hex or Dec) the excess will be ignored.

In general, when in doubt, type X. That will usually help. DO NOT USE CURSOR UP to get back into the code from the prompt window. Type X's and returns instead.

2.4.5 THE CONVERT DECIMAL FUNCTION - K

Use the f8 key to get back to $6000. Try typing a K. The prompt, CNVRT DEC and the blinking cursor appear in the prompts window. If you give this prompt decimal number 255 or less, you will see the number converted to Hex and entered in the address where the cursor had been. The display will scroll to the next address. A number between 256 and 65536 will be loaded into two bytes, and the screen will scroll two address locations. The two-byte converted number will be entered in 6502/Z-80/8080, etc. format, with the low-order byte at the lower address. Try converting 255. Don't enter the # sign first, and be sure to <RETURN> after your entry. You'll see FF entered. Try converting 40960. You'll see 00A0 entered. Try 65535 and you'll get FFFF. Try anything invalid, and you'll get an abort. Again, don't enter the # sign first, and be sure to <RETURN> after

your entry. Get back to $6000 when you are finished.

2.4.6 A WORD ABOUT THE SOUND EFFECTS

Notice that you get a boop noise every time you call for one of the special keystroke options of the editor. In general, the noises made are "boop" for editor options, "beep" for the first digit of a hexcode entry and "bip" for the completion of any entry or task. You are probably already aware of the noise it makes if your entry was none of the above or if an abort occurs.

2.4.7 MOVING BLOCKS OF CODE - T

After you've gotten the display back to $6000, type a T on the VIC keyboard. Notice the prompt "XFER FROM" and the blinking cursor appear in the prompts window. The T command is the means by which you can relocate blocks of code. The code is always moved TO the area the screen is displaying. In other words, after the move, the first address of the block of code that is moved (copied) will become the address of the line on which the blinking cursor appears. Try this example:

1. Type an X to get you back up into the code listing.

2. Scroll to address $600A and enter recognizable nonsense, such as AA, BB, CC, DD, EE.

3. Call $6000 back by using f8.

4. Type T.

5. Answer the prompt XFER FROM with $600A <RETURN>.

6. Count the number of nonsense bytes to be moved - in this case, five.

7. Add five to $6000. This equals $6005.

8. Type $6005 <RETURN> to the STOP@ nnnn prompt. The code should be copied in its new location.

Notice that your "original code" still can be found at $600A+. Once again, STOP@ defines the LAST address in Hex INTO which you will permit the transfer to occur. The address from which the blinking cursor was taken is marked by an asterisk and is now the FIRST new address of the block of code you're going to be copying. Note that STOP@ must be set to a HIGHER address than the current edit address, or the transfer will abort.

Do this next example. It will be used to demonstrate the Search and Remap functions. Locate yourself at $6000. Type T and answer the XFER FROM prompt with $1B5B <RETURN>. Answer the STOP@ prompt with 6220 <RETURN>. Try this procedure again. This time when you press <RETURN> after giving the address (which you can also do with MARKERS or in decimal preceded by #), you'll see 6220 reappear after the STOP@ prompt. The program remembers the last STOP and gives you the chance to reuse it just by typing the <RETURN>. If you want to change the STOP, just type the new address over the old. It will be given back to you the next time you are prompted for a STOP. You can abort the transfer by the same means that abort the DISPLAY WHERE? functions (type X), and if you decide to abort after getting all the way to the STOP@ prompt, you can still do so by Xing out the STOP address and hitting return.

Make sure that you have done the preceding transfer correctly before you proceed.

At this time, your screen should be indicating the EDIT HEX mode with $6000 as the center address. You will have transferred up to the PROMQUEEN RAM from the VIC's BASIC RAM the section of the Hexkit 1.0 program that manages the printing to the screen of all the prompts, logo, etc. If you scroll around the address range from $6000 to $601E, you will be looking at a 6502 machine code program that is called as a subroutine. It manages the printout of messages which are numbered from 0 to 26 in hexadecimal. When the message number is put into the accumulator with an LDA instruction, and the routine is called with a JSR instruction, the specified message is printed. The messages themselves are stored as character strings which you can see starting at $601F. Start

scrolling past $601F and you will see the messages scrolling by in the fifth column of the display. Notice that each message ends with a zero byte so the routine knows where it ends. Some characters are not indicated in the fifth column. These are spaces, control codes, color codes, etc.

We can run this routine, but it will not work at address $6000 because it was written to run at $1B6B. To make it work at $6000, we must "remap" it. No problem here.

2.4.8 SEARCH FUNCTION - U

Get the display back to $6000 (remember the f8 key?). Hexkit 1.0 gives you two different ways to search through memory to find specified bytes. Let's use the keystroke U first. Type a U. You will observe the following pop up in the prompts area:

```
SEARCH 20,*9,*C,*D,*E
```

The cursor will be blinking on the 2.

This is another one of those remembers-the-last-entry functions. When you press <RETURN>, the program will start searching the code from the cursor edit point to find the next byte in a higher address that is a 20 (in Hex), or any other byte that ends with 9, C, D, or E. This will catch all 6502 opcodes which may have to be remapped. The asterisk means "don't care." You can put an asterisk in the high-order nibble, the low-order nibble, or both. If you typed over this prompt so it becomes

```
SEARCH 20,**,**,**,**
```

the function will only look for "20" bytes. You can SEARCH for any five individual opcodes or operands you specify, and these five bytes will be remembered every time you use the U function. The program will stop on the first one of the five it comes to, regardless where the byte is located in the search command string. You can also abort: type some Xs and hit return to abort. Leave it as is for now and hit the <RETURN> key.

As soon as you hit <RETURN> you will see the display relocate to $6003. The "BD" in $6002 is one of the 6502 opcodes with an operand which may have to be changed when you are remapping. This time it does. The display stops on the first byte after such an opcode to facilitate your use of the remap function in Hexkit 1.0.

2.4.9 REMAP FUNCTION - R

Type an R. The prompt "REMAP:ADD" appears with a number in Hex (probably 0000 now). The cursor will be blinking on the first digit. This is another remember - the - last - value function. You have to specify what offset you want to add to the operand to remap the code. Calculate the offset you need as follows:

```
    new location of routine
  - old location of routine
  -------------------------
  = offset required
```

In this case it is

```
    $6000   (You must do this in Hex)
  - $1B5B
    -----
  = $44A5
```

Type over the 0000 with 44A5 and hit <RETURN>. The operand has been remapped, and the display scrolled down two locations.

If you ever want to abort this, X out the offset and hit <RETURN>.

Two things to look out for:

1) SEARCH function U may find a lot of bytes that don't need to be remapped. In your notes on your routines, you can keep track of those commands which will need remapping if the code is relocated.

2) Sometimes you will be remapping downward in the memory space. To get the right offset, first add $10000 (ten thousand Hex) to the new location of the routine before you do the subtraction above.

One more operand must be remapped before this routine will work at $6000. Use the U function again. Now the display should be relocated to $6009. This is another BD (load the accumulator absolute indexed by X) instruction, and must be remapped. Hit the R key. You'll see your offset of 44A5 remembered--it is still good. Hit the <RETURN> key, and you will have completed remapping the message printer routine.

2.4.10 SEARCH FOR SPECIFIC ADDRESS - Q

Remember that the message routine knows where messages end by detecting a zero byte, but it must look up where the messages start in a table. The start of the table can be found by using the other search function in Hexkit 1.0. The first address listed in the table we know from our notes. It is $1B7A (in Hex). Type a Q. See the prompt SEARCH and the (probably) 0000. Type over the zeroes with $1B7A and hit <RETURN>. We have found the start of the table. The cursor is blinking on the low-order byte. This address needs to be changed. Type an R and <RETURN>. Unless you changed it, the offset we calculated is still there. Type R <RETURN> decimal 38 more times to remap this particular table. (Q has done its work--you don't need it anymore.) If you ever want to abort a Q search when you have selected it by mistake, X out the number after the prompt and hit <RETURN>.

The routine is ready to work now at its new location, but we must give it message numbers before we call it. Let's enter the necessary routine at address $6400. Use the f8 key to move the display there. The routine below will drive the message routine to print the first 10 (in Hex) messages from the message table.

M	ADDRESS	6502 ASSEMBLY FORM	6502 MACHINE CODE
	6400	LDA#$00	A9 00
M1	6402	PHA	48
	6403	JSR$6000 (message)	20 00 60
	6406	PLA	68
	6407	CLC	18
	6408	ADC01	69 01
	640A	CMP#$10	C9 10
	640C	BCC M1	90 - (F4)
	640E	JMP$0FB (get in)	4C FB 10

Starting at address $6400, type in the codes A9 and 00. These two bytes tell the 6502 to put zero into its accumulator. Before you type the next code (48, which pushes the accumulator onto the stack), mark the location by typing an M and then a 1. You will see that this location has been labeled M1. We did this to allow Hexkit to figure how to branch back to here later.

Now enter the 48 hexcode, and proceed through the rest of the routine until you have typed the 90 instruction into location $640C.

We stopped at $640D because the routine contains a loop which is terminated with a conditional branch instruction at address $640C. Hexkit will help you find the necessary offset to $6402 because we have MARKED the location to where the branch must go. Now you can use the OFFSET function of Hexkit.

2.4.11 THE OFFSET FUNCTION - O

Type an O. You will get the prompt:

LD OFFST M

and a blinking cursor. Type the marker number, which is 1. Then you will be asked

PUT H OR T

meaning "put the offset HERE (at the current edit address) or THERE (where the marker was assigned)." This is a branch back, so we need the offset HERE. Type an H. Anything typed other than H or T aborts the function. You could also have aborted earlier by typing anything other than a Hex number after the M. The function aborts automatically if you have asked it to calculate too long a branch. This time it should work; the number you see entered is the proper signed two's complement form. Go ahead and type the rest of the program (4C FB 10).

2.4.12 BACKWARD AND FORWARD BRANCH OFFSETS

You will find that branch instructions may occur forward or backwards. Backwards branches work just as the example we performed above. To use Hexkit on FORWARD BRANCHES, mark the location where the offset must go, leave a space for it (enter FF in the location, or tap the f7 key) and continue entering the program until you come to the statement to which the branch must occur. Then use the O keystroke and tell it to put the offset THERE by typing a T.

2.4.13 DEBUGGING AID FUNCTIONS - P, J, N, L

Remember that all functions are summarized for quick reference on pages 54 to 56.

We can use the debugging aids to check the routine we have been working on. Scroll the display back to the beginning (at $6400) and start scrolling down. At the low-order byte of the operand of the JSR instruction (the display at $6404), stop scrolling.

WHEN YOU TYPE A P the location specified by the operand will be displayed. In other words, the P command enables you to look at where a subroutine, etc. are going to be. If this were not the location we really wanted, we could now scroll around until we found the proper location, and then tag it with a MARKER.

THE J COMMAND For our example, tag address $6000 with M0. Now, type a J, for jump back. The display returns to just where it was before you typed a P.

TYPE AN N to generate the prompt LD ADDR M and the cursor. Type the zero denoting M0. You will see that the address you tagged with M0 has been entered and the display scrolled two locations. Notice that the address is entered in 6502, Z-80, 8080, etc. format, low-order byte first.

THE L COMMAND Keep on scrolling and checking until you get to the offset we calculated for the BCC (90) instruction at address $640D. Type an L (for look at) and the display will relocate to the destination of the branch, which should be the 48 (PHA) instruction at $6402. If this were not correct, we could scroll around to tag the correct location with a marker; reenter the correct offset after we bad got back to its origin by using the J (Jumpback) key; then the O key for offset entries.

Keystrokes J, L, and P are all instantaneous; you can't abort, but you can always get back where you were by typing J. The J function also works after the f8 key (relocate display).
Once you have checked the routine at $6400, you can run it from EDIT HEX. Scroll (or f8) the display to put the cursor on the first byte of the routine you want to test. Put the display at $6400. Mark the location for convenience with MA.

2.4.14 **INITIATE MACHINE CODE PROGRAMS** by typing an I at the cursor location. Be sure the cursor is blinking on the first byte of the routine you are trying. To see if our example program works, put the display on $6400 (its entry point) and type an I. Now you should see the first

16 prompts printed to the screen. We added the call to the routine "get-in" to our routine to freeze the action. Type any key to see the normal EDIT HEX display at $6400.

*** NOTE THAT all text so far is based on an ongoing lesson. All addresses and markers are geared to one specific routine.**

If you are not successful at this point, go back and try once more. You need the running routine to carry out the rest of the sample routines. Move to address $6500 for a demonstration of additional useful functions.

2.4.15 C/MODE FOR AUTOMATIC VIC ASC CONVERSIONS

After moving to $6500, SPECIAL FUNCTION key f1. C/MODE appears in the prompts area, and the cursor is in the fifth column of the display to indicate the program is ready to enter the VIC ASC value in Hex for any key except the special function keys. In C/MODE, type your name. Watch it be entered in Hexcode letter by letter. Notice you can also type the cursor controls, color controls, reverse on, in fact any key on the main body of the keyboard for which an ASC value exists, and it will automatically be converted to code and entered. This makes it easy to make message tables for all kinds of displays. Hit A again to get out of C/MODE.

When in C/MODE, you can insert a zero byte to indicate end of message just by typing f3. (This key is disabled in normal mode.)

You can tag the beginning of your messages with a marker when in C/MODE by typing f6. You then give the marker number in the usual way.

In addition, you can insert and delete spaces. To insert a space at the current edit position, type f2. You will be prompted with the message INSERT and the current STOP@ location, this defining upper boundary of the list of code which will be shifted to open a space. It must be greater than the current edit location or the function aborts. To abort intentionally, X out the STOP address and press <RETURN>.

To delete, press the f4 key. You will see the prompt DELETE and the current STOP@ address defining the upper limit of the list to be shifted down to delete. X out the STOP@ address and press <RETURN> to abort. You don't need to reenter the STOP@ address each time; just hit <RETURN> if it is the one you want to keep using. INSERT and DELETE also work in normal mode.

C/MODE toggles on or off each time you press f1. Try it a few times and see.

Now, you should experiment. Try revising the Hexkit prompts list you have made. You can even change colors. If you are in C/MODE, the ASC for the key will automatically be entered. Try making your own custom prompts table by generating your prompts in C/MODE. Be sure to mark the start of each prompt by using the f6 key when in C/MODE, and end it with a zero byte by typing f3.

If you use the markers in numerical sequence, you will find that you can easily unload the marker addresses into a table by repeated use of the N keystroke. NOTE: Remember that no lettered keystroke command works when you are in C/MODE. Use markers to redirect the operands of the two BD opcodes at $6002 and $6008, which we remapped a few pages ago, to the base of your message address lookup table. You can use markers and the N function to do this again. Watch the routine at $6400 print out your message table. Take the time to practice. It rapidly gets easier. All the keystroke commands are summarized in the summary section, but here is a quick review list of SPECIAL FUNCTION keys:

f1 - enable/disable C/MODE
f2 - insert space
f3 - 00 byte for end of message
f4 - delete
f5 - scroll to next lower address
f6 - C/MODE (or other) marker for beginning of message
f7 - scroll to next higher address

2.4.16 THE STUFF FUNCTION - S

One last keystroke remains to be explained. The S key calls the STUFF function. This will stuff all memory locations from the current edit position on up to the address of the STOP@ location with any byte you specify. Just check that the STOP@ is where you want it, type over it to change, or X it out and press <RETURN> to abort. Once you have pressed <RETURN>, you will be asked for the byte to stuff with. Give it the byte you want or 2 X's to abort, and press <RETURN>. Be careful with this one! It works very fast.

GENERAL COMMENTS ABOUT THE HEXKIT EDITOR

The Hexkit 1.0 Machine Code Editor is a useful general purpose tool intended to make the VIC-20 equipped with a PROMQUEEN quite serviceable for 6502, Z-80, 8080, 8085, etc. machine code development; however, be careful if you are coding for a 6800, 6809, or 1802. We think that you will find it a little harder to hand code your programs than to use an assembler, but if your documentation is thorough, you will probably come to find it much easier to debug routines written in code than to edit with an assembler. Take a little time to practice now. Your efforts will be rewarded. When you have satisfied yourself with EDIT HEX, just type an X or two to get back to the main menu.

2.5 ANOTHER ADDRESSING REVIEW

Before discussing the Hexkit 1.0 BURN EPROM ROUTINE, option B on the Hexkit 1.0 Menu, let us review the PROMQUEEN hardware one more time.

The PROMQUEEN cartridge occupies a continuous 8-kilobyte block of address space. You select which of four possible locations these 8 kilobytes will occupy when you set the BLOCK Select DIP switch. The table on page 52 of this manual tells you what range of addresses the PQ will recognize if you set the PROMQUEEN RAM and the EPROM SOCKET into any of the four possible expansion blocks. Study the table carefully. The PROMQUEEN simply ignores whatever the VIC microprocessor is doing unless it happens to be either writing to or reading from an address from the BLOCK the PROMQUEEN is occupying; therefore, you must know what range of addresses to use when you want to communicate with the PROMQUEEN RAM or the EPROM SOCKET.

Within the PROMQUEEN, the 8K of address space is further subdivided into two halves. The first half (lower numbered addresses) is devoted to the 4-kilobyte PROMQUEEN RAM. The second half is devoted to EPROMs of up to 4 kilobytes of memory capacity. The PROMQUEEN RAM and EPROM never occupy the same address space, and they are not bank switched. If the cartridge is in the normal PROG mode, you can uniquely address any single location out of all the decimal 8192 locations, and any single location is always addressable. This is why you usually have to remap working VIC code in the PROMQUEEN RAM if you want to BURN it on EPROM and have the EPROM able to work from the ZIF SOCKET. Remember, you don't have to remap code for external computers tested in the target EPROM socket when you are using a jumper cable and the cartridge's MIMIC mode.

2.6 THE HEXKIT 1.0 BURN EPROM ROUTINE

The PQ RAM models exactly what will be burned onto an EPROM by the Hexkit BURN routine. You first set up a copy in PROMQUEEN RAM of the EPROM you want to make; you then switch the cartridge to BURN mode, and you answer

prompts with the starting and ending addresses on the EPROM into which you want to copy PQ RAM address contents. If you tell the cartridge to burn addresses (start address of EPROM) + 5 to (start address of EPROM) + 25, it will find the code in the range of addresses from (start address of PQ RAM) + 5 to (start address of PQ RAM) + 25. The address FROM which the code burned on EPROM is fetched is always the address where the code is to be burned minus Hex $1000 (decimal 4096).

When you set the cartridge to BURN, you isolate its data bus from the VIC. The cartridge is placed into an altered logical condition in which it looks for the VIC attempting to write data into the EPROM in the ZIP socket. Whatever the VIC writes doesn't matter, because the PROMQUEEN data bus is disconnected from the VIC. Instead, the cartridge will freeze the address to which the write is made, the same address will be applied to both the PROMQUEEN RAM chips and the EPROM, and then a programming pulse long enough to burn the data coming out of the PQ RAM onto the EPROM (50 milliseconds) is generated. Care is needed not to feed writes to the EPROM too rapidly, or addresses will be missed while the PQ remains frozen on the previous address it detected.

BURN EPROM takes care of this. It also checks the EPROM to make sure it is erased before attempting the burn, and it VERIFIES that what ended up in the designated range of addresses on the EPROM is in fact an exact copy of the corresponding address range in PQ RAM.

We will proceed as though we were going to burn the program we wrote while working with EDIT HEX. (It doesn't matter if you've obliterated it.) We won't remap, but we would need to if we wanted it to work directly from an EPROM in the ZIF SOCKET.

2.6.1 THE BURN PROCESS - - - B

When in EDIT HEX normal mode, you can return to the Menu by typing an X or two. Select BURN EPROM by typing a B. As soon as you do this, you will see the message BURN EPROM, the prompt FROM, and a blinking cursor. You can respond with an address in Hex, decimal, or use a MARKER

number. Typing an X returns the Main Menu. Invalid input results in a "psst" noise and no progress.

GIVE THE ADDRESS ON THE EPROM where you want the burn to start. (remember that what should be burned there is the PROMQUEEN RAM code address plus $1000.) Since our program starts at $6000, we need to answer $7000 to FROM. Type in $7000 and <RETURN>. The prompt, TO, is asking the address where you want the burn to end. If we burn to $7500, we'll catch everything we wrote up to address $6500 in PQ RAM. Again, MARKERs, Hex numbers preceded by $, and decimal numbers preceded by # sign will be accepted. Abort by typing an X. If you want to change where the burn starts, give any invalid entry. Hit <RETURN>. If you've given the TO a lower numbered address than the FROM, you'll be given the prompt again.

Before the BURN, Hexkit checks to make sure an erased EPROM is in the socket. Since Hexkit is there, the screen is probably black, and you are being told to "ERASE EPROM."

```
**********************************************
At this time, turn the socket OFF and REMOVE
the Hexkit EPROM from the ZIF socket.  You
do not want to damage your HEXKIT by mistake.
**********************************************
```

Now, <RETURN>. The go ahead to burn has been allowed, because there is no EPROM in the socket and the socket is OFF. In a real situation, at the ERASE EPROM display, you can change the EPROM to one that is erased. If you do this, the FROM:TO information does not need to be reentered. Just turn the socket OFF, change EPROM, and turn the socket back ON. As soon as you type <RETURN>, Hexkit 1.0 will check the new EPROM to make sure FF is in every location in the range you have told it to burn. (You can abort back to the Main Menu by typing an X.) When the erasure check is finished, a "clean" EPROM causes the SET SWITCH TO BURN display. Set the PROG/BURN switch on the PROMQUEEN cartridge to BURN. (The MIMIC/PROG switch must be set to PROG and the socket must be ON as indicated by the green LED.)

When you flick the PROG/BURN switch away from you, the 25

volt power supply fires up to burn the EPROM, and the cartridge starts looking for writes to the address space occupied by the EPROM. Nothing will be burned on the EPROM until you hit <RETURN>, but don't <RETURN> before you switch to BURN, or you will miss some addresses.

We don't suggest that you actually burn an EPROM with the message routine we remapped. Therefore, don't switch to BURN, just bit <RETURN>. Nothing would happen to the EPROM under these circumstances, but you will see the BURNING ADDR display, color TV screens will be red, and ADDR will continually be updated with the address that would have been burned on the EPROM. This will continue for as long as it takes to burn the range of addresses you specified, and can be up to five minutes for a whole 2732 or 2732A. This is probably the only Hexkit function that takes much time.

When the burn is finished, you will see the screen change again (blue) and you will be prompted to turn the PROG/BURN switch back to PROG. You must do this to obtain a valid error check because no data can pass from PROMQUEEN to VIC in BURN mode. (If you were to type <RETURN> before you set the switch back to PROG, you'd see an error indication and the display would freeze telling you about an error at the first address you attempted to burn.) If the burn is good, the screen turns green almost instantly. If there is an error, the screen will remain blue to tell you about it. Hit <RETURN> again to see the next error location. Hit an X when you have seen enough errors; if one address is bad, they all are. The X will return you to the Main Menu.

Remember: you must give the addresses on the EPROM that define the lower and upper limits of the range you want to burn. If you give some other address range, but set the switch to burn and try to proceed, nothing will happen to the contents of the EPROM (unless you have the EPROM type switch misset). The cartridge must be set to burn and Hexkit must feed it addresses on EPROM for anything to actually be recorded.

2.6.2 BURNING EPROMS FROM BASIC

You can use a BASIC one-liner in immediate mode as a BURN program.

When you set the PROMQUEEN BURN Mode (SOCKET ON; MIMIC/PROG in "PROG"; EPROM set to EPROM type desired and PROG/BURN to the BURN position), the programming power supply will come on and the PROMQUEEN RAM will become write protected against the VIC (you won't be able to read the PROMQUEEN RAM from the VIC either). Logic on the PROMQUEEN circuit board is set to examine the address and block lines coming from the VIC for any **writes** to the address space occupied by the PROMQUEEN EPROM socket. When the cartridge sees a **write** to an address on EPROM (remember this changes when you select a different expansion block setting for the cartridge), it duplicates the contents of a corresponding location in PROMQUEEN RAM onto the EPROM. The corresponding location in RAM from which the byte for burning is fetched is **always** the address on EPROM to which the write is made **minus** decimal 4096 ($1000). Burning takes about 70 milliseconds per byte, so a delay must occur between burning of successive bytes. The required BASIC syntax to do this is:

```
FOR I=0 TO L-1:POKE A+I,0:FORK=0 TO 65:NEXT:NEXT
```

I and K are indexes. L is the number of bytes you want to burn. A is the start location on the EPROM in the VIC's address space where you want the burn to start. You will have to use the actual numbers for A and L to enter them into BASIC's immediate mode. Make sure the cartridge is already set into burn mode before you hit the return key after typing the one-liner.

REMEMBER

> This routine won't check erasure before the burn or verify it after. Hexkit's BURN Routine will.

2.7 SAVING, LOADING, AND VERIFYING TO DEVICES

Hexkit 1.0 provides some twists on the usual VIC load, save, and verify commands to facilitate handling of blocks of machine code. It also supports the disk drive.

Whenever you type options C, D, or E on the Hexkit Menu, your acknowledgement is the routine name. You will always be asked for the device number, which can be 1 for the cassette, or 8 for the disk. You can also type an X to abort. You are always prompted for a program name. This can be omitted if it is a communication with the cassette; a <RETURN> is sufficient. Keep your names under 16 characters and <RETURN> after the name.

Remember to put a formatted disk in your disk drive and initialize it in the VIC's immediate mode before you attempt disk operations through Hexkit.

On SAVES, you are always asked FROM and TO. You can specify what code to save this way. You can do it with MARKERs, # decimal, or $ Hex responses. The code will be saved under the name you gave it, a reload header will be affixed to the file, and if it's saved to the disk, it becomes PRG file. When the SAVE is complete, you'll see the Menu, otherwise you see a Kernal error message, such as "IO ERROR #5" which means device-not-present. Kernal error messages are listed on page 210 of the VIC Programmer's Reference. Typing any key brings back the Menu. You cannot SAVE out of BLOCK 5, but you can relocate load there - we'll get to that.

BASIC programs can be stored with Hexkit's SAVE routine provided you include in the range of code you save a little code program that resets the BASIC pointers (locations $2B to $38 on page zero). SYS the pointer resetter routine before you try to run the BASIC after a reload. Remap your Hexkit up to high RAM and use the SAVE routine to store mixed BASIC and code programs this way. You will find it much easier to type in code with Hexkit than to enter it through BASIC with DATA statements. These mixed language SAVEs reload properly with the VIC's own LOAD command; the only trick is to start execution by SYSing to the code routine you put

there to reset the pointers. Be sure to save mixed language saves from the start of BASIC. (You find this vector in address pair ($2B-2C).

After you SAVE, you can verify by typing E on the Menu. Rewind your tapes first. You can type an asterisk and then <RETURN> in response to the prompt NAME if you are verifying a disk save you just made, or omit the name if it was a cassette save. You will see the usual Kernal messages about device status. You'll see OK if it passes and ERROR if it doesn't. The cursor will be blinking at the conclusion of the test and you get back to the Menu, then, by typing any key. You cannot put an end-of-tape marker on tapes through Hexkit.

The Hexkit LOAD FROM DEVICE runs after you type a D to the Main Menu. It will give you the prompt NEW LOC? You can tell Hexkit to put the code somewhere other than the location specified by the header by typing a Y. Any other key will do to say NO, and X will take you back to the Menu. If you say "yes," you will be asked WHERE?: Marker, $Hex, or #decimal answer are acceptable. Then you give the name. The usual Kernal messages will report I/O errors.

If you don't say yes to NEW LOC?, the prompt WHERE does not occur.

The Menu is restored when a successful load occurs. Otherwise, type any key for the Menu after you've digested the error message.

Most programs don't work after a relocate load: they need to be remapped. But you can do such things as use Hexkit to shift down code from ROMs in BLOCK 5 to RAM in some space you CAN save from; put the code on some device, and later relocate load it to RAM you have put into BLOCK 5.

END OF HEXKIT TUTORIAL

3.1 PROMQUEEN CARTRIDGE User Controls

The notes below refer to the drawing on the opposite page.

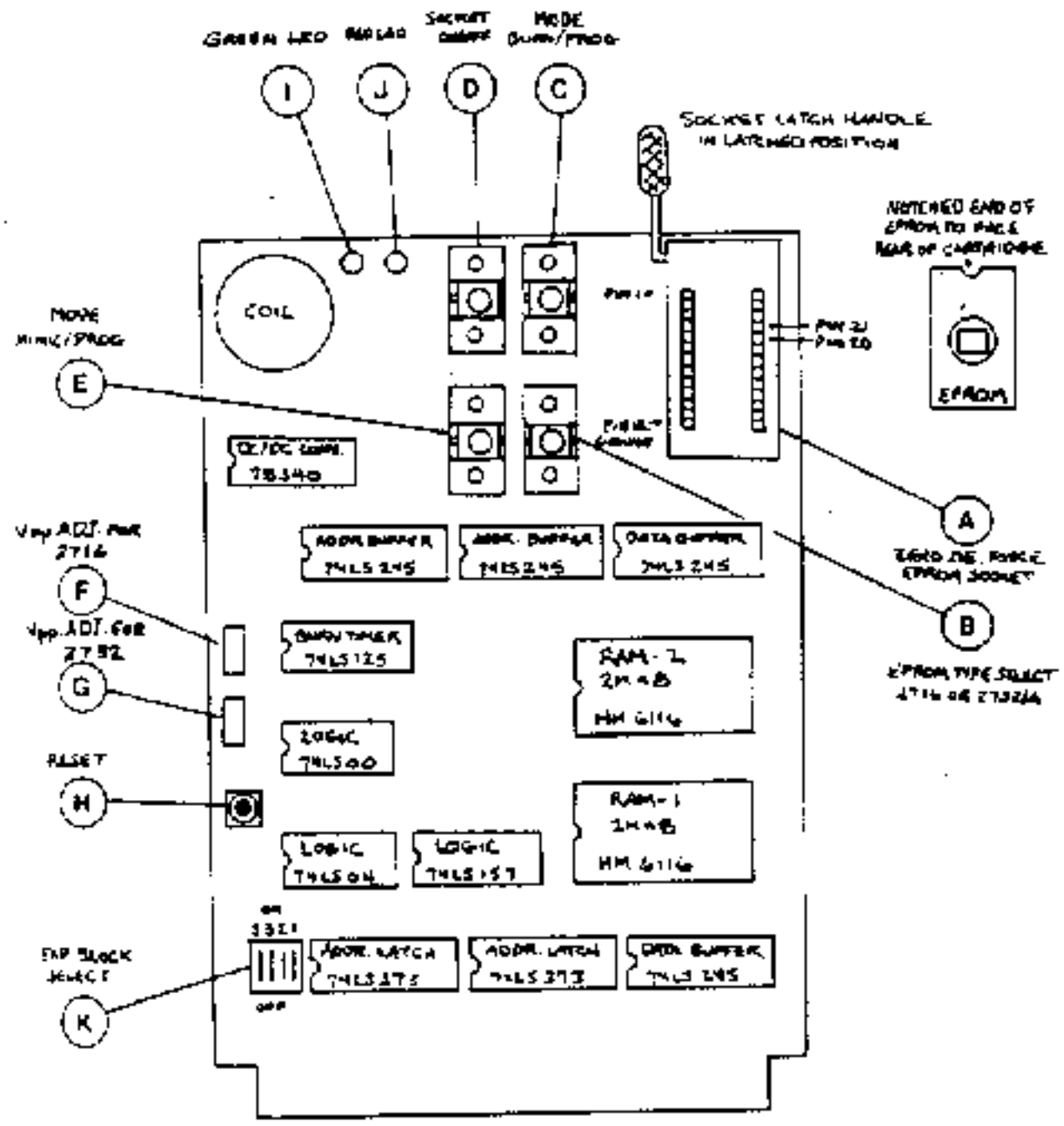
A) **ZERO INSERTION FORCE (ZIF) SOCKET.** Release the latch handle to install or remove EPROMs. Note that the notched end of the EPROM must face away from you. Always disconnect the SOCKET by flipping switch (D) away from you when removing or installing EPROMs. Connect a 24-pin DIP jumper cable between here and the target ROM socket when using the PROMQUEEN in MIMIC mode.

B) **EPROM TYPE SELECTOR SWITCH.** Flip this switch away from you to use or mimic 2716 EPROMs. Flip this switch towards you to use or mimic 2732A EPROMs. Always make sure this switch is properly set before you burn an EPROM. If this switch is set for the wrong EPROM type when you attempt to burn, that EPROM may be destroyed. **NEVER USE ANY OTHER THAN 2716 OR 2732 OR 2732A EPROMS WITH THE PROMQUEEN UNLESS YOU KNOW WHAT YOU'RE DOING.**

C) **BURN/PROG MODE.** This switch applies the programming voltage to BURN EPROMs. Always leave this switch set to PROG (vertical) unless you are using the Hexkit BURN routine which **WILL TELL YOU WHEN TO FLIP THIS SWITCH TO BURN.** When the BURN voltage is applied, the **RED** LED will come on, indicating that the DC/DC converter is working. You may hear a faint buzzing noise. This is completely normal.

D) **ZIF SOCKET CONTROL SWITCH.** This switch removes all voltages from the ZIF socket when flipped away from you. When the green LED goes out (switch off) you can change EPROMs or connect the MIMIC cable without having to turn off the VIC. This switch must be in the ON position (upright) to access EPROMs, BURN them, or use the MIMIC mode.

E) **PROG/MIMIC SWITCH.** Leave this switch set to PROG (flipped towards you) unless you are using the MIMIC mode. The green LED will go out when you flip to MIMIC. In the MIMIC mode any programs you have entered to the PROMQUEEN RAM are available thru the ZIF SOCKET to an external microprocessor, and the PROMQUEEN RAM is



WARNING Use the PROG/BURN switch only to start or terminate a burning run. Change of other switch settings during a burn could damage the EPROM.

CIRCUIT BOARD LAYOUT AND USER CONTROLS POSITIONS

disconnected from the VIC. If the EPROM Selector Switch is set for 2716, only RAM-1 is accessible. If the EPROM Selector Switch is set for 2732As, both RAM-1 and RAM-2 are accessible to the external computer.

F) **2716 PROGRAMMING VOLTAGE ADJUSTMENT POT.** This is set at the factory so that the DC/DC converter makes 25.0 volts when BURNing 2716 EPROMS. To measure the 2716 programming voltage, remove any EPROMS from the ZIF socket, set the PROG/MIMIC Switch to PROG, the SOCKET switch to ON, the EPROM type selector switch to 2716, and **last** the PROG/BURN switch to BURN. Measure the voltage between pin 21 and the ground pin (pin 12) of the socket with a VOM of impedance of 10,000 Ohms/V or greater. **FLIP THE BURN/PROG SWITCH BACK TO PROG BEFORE YOU OPERATE ANY OTHER SWITCHES, AND WAIT UNTIL THE RED LED GOES OUT.**

G) **2732/2732A PROGRAMMING VOLTAGE ADJUSTMENT POT.** This is set to 25.0 volts at the factory. 21 volts is the programming voltage for 2732A EPROMs. You can use the PROMQUEEN to program 2732A's by readjusting the pot to provide 21 volts. The procedure is the same as given above except that the EPROM type selector switch is set to the 2732 position, the voltage is measured between pin 20 and ground, and you adjust the 2732 pot, not the 2716 pot. **NEVER ATTEMPT TO PROGRAM A 2732A EPROM WITH THIS POT SET FOR THE 25V 2732 PROGRAMMING POTENTIAL.**

H) **RESET BUTTON.** If you crash the VIC with a faulty machine language program in PROMQUEEN RAM, you do NOT have to turn the VIC off (which would erase your program from the PROMQUEEN RAM). Just tap the reset button. You can restart Hexkit by typing SYS 4112 <RETURN>.

I) **ADDRESS SELECTOR DIP SWITCH.** Allows the cartridge to occupy any of the 4 expansion blocks of the VIC. **DON'T SET MORE THAN 1 BLOCK AT A TIME.** (See page 54 for the addressing table.)

NOTE: The VIC operates at a clock rate of 2 MHz. To be compatible with the VIC, EPROMs must have an access time rating no longer than 450 nano seconds.

3.2 PROMQUEEN ADDRESSING

COMMANDS TO LOAD HEXKIT PROGRAM INTO VIC RAM	EXPANSION BLOCK USED	LOCATION OF PROMQUEEN RAM IN VIC ADDRESS SPACE	LOCATION OF EPROM SOCKET IN VIC ADDRESS SPACE
SYS12619 RETURN RUN RETURN	1	START ADDRESS HEX:2000 DEC:8192 LAST ADDRESS HEX:2FFF DEC:12287	START ADDRESS HEX:3000 DEC:12288 LAST ADDRESS HEX:3FFF DEC:16383
SYS20829 RETURN RUN RETURN	2	START ADDRESS HEX:4000 DEC:16384 LAST ADDRESS HEX:4FFF DEC:20479	START ADDRESS HEX:5000 DEC:20480 LAST ADDRESS HEX:5FFF DEC:24575
SYS29039 RETURN RUN RETURN	3	START ADDRESS HEX:6000 DEC:24576 LAST ADDRESS HEX:6FFF DEC:28671	START ADDRESS HEX:7000 DEC:28672 LAST ADDRESS HEX:7FFF DEC:32767
SYS4544 RETURN RUN RETURN	5	START ADDRESS HEX:A000 DEC:40960 LAST ADDRESS HEX:AFFF DEC:45055	START ADDRESS HEX:B000 DEC:45056 LAST ADDRESS HEX:BFFF DEC:49151

NOTE 1 Addition of RAM to BLOCK 1 causes the VIC Kernal to relocate the screen memory and BASIC program space in the VIC's RAM. If you have set DIP switch #1 on the PROMQUEEN, you will see upon start-up that you have 7679 bytes free before you load Hexkit. You can see use PROMQUEEN RAM for BASIC program storage when not using the PROMQUEEN with HEXKIT when the PROMQUEEN is set to operate in BLOCK 1. LOADING Hexkit CAUSES SCREEN MEMORY TO BE RESET AS THOUGH THE VIC WERE UNEXPANDED.

NOTE 2 The Kernal does not allow saving programs to devices from out of BLOCK 5. This is to prevent users from copying program cartridges to tape. Hence the Hexkit SAVE routine does not work when the PROMQUEEN is set to operate in BLOCK 5.

NOTE 3 Whenever you use the RESET button the VIC will check memory to see where it must put the screen and program memory. If you use RESET with contiguous expansion RAM at $2000 or above then HEXKIT will be damaged by the RESET. You must load HEXKIT back off its EPROM to run it again.

GENERAL NEVER SET MORE THAN 1 OF THE DIP SWITCHES AT A TIME. THIS CONFUSES THE VIC.

3.3 HEXKIT 1.0 KEYSTROKE COMMANDS IN EDIT HEX MODE

I INITIATE

is used to start the machine code routine beginning at the byte located under the blinking cursor in the display.

J JUMPBACK

to display the address which had been displayed before you used the L, P, or f8 keys.

K CONVERT

single and two-byte decimal numbers to Hex and enter them. 2-byte numbers are entered low-order byte first.

L LOOK

at the destination of an offset under the blinking cursor. Jumpback by typing J to return.

M ASSIGN A MARKER

numbered from 0 to F in Hex to the address at the blinking cursor.

N LOAD THE ADDRESS

at which the designated marker was placed to the location of the blinking cursor, low-order byte first.

O FIND OFFSETS

to and from markers, and enter at either end of the jump, one end defined by the blinking cursor, the other by the Marker.

P POSITION

the display at the location whose low-order byte is presently displayed under the blinking cursor. Use J to return.

Q SEARCH

for a specified 16-bit operand. Put the display there.

R REMAP

by adding a specified offset to the 16-bit operand whose low order byte appears under the cursor.

S STUFF

a range of bytes from the blinking cursor up to the designated stop address with a specified byte.

T TRANSFER

a block of code starting at a designated address to the range of addresses which begin at the blinking cursor and end at the designated stop address.

U USER-DEFINED OPERAND SEARCH

put the display on the next byte after detecting a match.

3.4 THE HEXKIT FUNCTION KEYS

f1 C/MODE on/off toggle

f2 Insert up to stop address

f3 If in C/MODE, insert a zero stop byte

f4 Delete up to stop address

f5 Scroll display to next lower address and bip

f6 Alternate set Marker keystroke that works in C/MODE

f7 Scroll display to next higher address and bip

f8 Reposition display to new address. Jumpback with J.

X Your general purpose abort key

CURSOR UP - CURSOR DOWN

Repeating display scroll

3.5 BURN ROUTINE SUMMARY

1. B to the Main Menu

2. FROM prompt
   Give lowest address of your code in PQ RAM + $1000

3. TO prompt
   Give highest address of your code in PQ RAM + $1000
   (DO NOT RETURN)

4. Upper left switch ON
   Green LED on

5. <RETURN>

6. Erase check
   Black screen: turn OFF socket,
   change EPROM, turn ON socket
   <RETURN> again

7. Message: SET SWITCH TO BURN
   Opper right hand switch on

8. Red LED on

9. <RETURN>

10. Burn now in progress. Wait.

11. (Blue screen), Message: SET SWITCH TO PROG
    Upper right switch OFF

12. <RETURN>

13. BURN VERIFIED = Good

14. ERROR AT nnnn = Bad

15. X to Menu, or <RETURN> to find the next error

4.1 USING MIMIC MODE

WARNING: MAKE SURE YOU DON'T TURN ON THE EPROM BURNER WHEN A MIMIC CABLE IS CONNECTED: YOU COULD DAMAGE YOUR EXTERNAL COMPUTER. THE BURNER IS INHIBITED WHEN THE PROG/MIMIC SWITCH IS SET TO MIMIC, SO LEAVE THIS SWITCH SET TO MIMIC WHEN A MIMIC CABLE IS INSTALLED.

MIMIC mode is provided so that you can use your VIC's keyboard and other program entry means to load programs for your own experimental computer devices into the PROMQUEEN RAM and access these programs from your own separate computer devices as though the PROMQUEEN RAM were a programmed EPROM in the EPROM socket in your experimental computer device. This capability is ordinarily provided only by "microprocessor development systems" which are usually many, many times more expensive than your PROMQUEEN Cartridge.

To use the MIMIC capability, first put the program you want to try into its proper location in the PROMQUEEN RAM (corresponding to the location you want it in a finished EPROM) by whatever means you choose. These programs are almost always in the machine language of the microprocessor in the external computer device. You can enter the program using Hexkit, copy it from an existing EPROM (either a 2716 or 2732) you already have and modify it as desired using Hexkit; write it (if it is for a 6500 series processor) using VICMON, load it from tape, disk or MODEM or the user port of the VIC--there are many ways to do it. While loading, the PROMQUEEN must be in the normal PROG mode and set for the type of EPROM to be mimicked (the green LED on, the red off). YOUR EXTERNAL COMPUTER DEVICE **MUST** BE EQUIPPED TO USE EITHER 2716 OR 2732 SERIES EPROMs IN THE SOCKET YOU WILL CONNECT TO THE PROMQUEEN. Set the PROMQUEEN EPROM selector switch accordingly.

Turn off the EPROM socket before connecting any cables to the PROMQUEEN ZIF socket. Flip the PROG/MIMIC switch to MIMIC. You can no longer access the PROMQUEEN from the VIC. Leave the VIC on--it is supplying the power to the memory in the cartridge.

Turn the socket switch OFF and remove any EPROMs in the PROMQUEEN ZIF socket. It's usually advisable to turn the external computer device off. The green light on the PROMQUEEN should be out.

Now connect a 24-pin DIP jumper cable to the ZIF socket on the PROMQUEEN and latch it down. TO AVOID A POSSIBLE SOURCE OF DAMAGE USE A CABLE THAT HAS PIN 24 (Vcc) CLIPPED OFF ON ONE END OF THE CABLE OR THE OTHER. THIS WILL PREVENT YOUR EXTERNAL COMPUTER DEVICE FROM POWERING THE DC/DC CONVERTER IF YOU SHOULD ACCIDENTALLY FLIP THE PROG/BURN SWITCH TO BURN. ALSO, NEVER LEAVE THE JUMPER CONNECTED DURING USE OF ANY OTHER THAN MIMIC MODE. IF YOU TURN ON THE BURN MODE POWER SUPPLY WITH THE CABLE CONNECTED, YOUR EXTERNAL COMPUTER COULD BE DAMAGED. Make sure you have no twists in the cable (i.e., that pin 1 of the ZIF socket connects to pin 1 of the target EPROM socket in the external computer device), and plug the other end of the cable into the target EPROM socket. Avoid cables over 3 feet long--they are more likely to pick up noise. (Ideally, the MIMIC cable is shielded. If you are using an unshielded cable and a routine that should work doesn't, try wrapping the cable with aluminum foil.)

Turn the PROMQUEEN SOCKET switch ON (the green light will stay out) to try your program on the external computer device. If it works--flip the SOCKET switch off, remove the cable, install an erased EPROM, flip the PROG/MIMIC to PROG, the SOCKET switch back on, and burn the program onto a compatible EPROM in the usual way. That EPROM will work in the target EPROM socket.

By the way, you don't need to burn any more bytes on the EPROM than those occupied by your debugged routine. You can build your programs onto the EPROM step by step.

The PROMQUEEN is made to MIMIC for 6502 type microprocessors which do not separately decode the EPROM's CS and OE signals. These 2 pins are shorted together inside the PROMQUEEN. Some 8080, 8085, Z-80, etc. systems do separately decode these signals. If so, you must clip off the OE pin on your jumper cable in order to prevent bus conflict which would cause the external computer to crash. Also some Atari ROM cards and the like short OE directly to ground. Clip off the OE pin on the cable to make MIMIC work properly through the PROMQUEEN.

4.2 REMAPPING HEXKIT WITH VICMON

VICMON has a very handy remapping function (keystroke N) which is a worthwhile addition to the search and remap functions in Hexkit. If you have VICMON, use it to disassemble Hexkit. You will find that it is organized so that the program itself occupies one continuous block of memory (from $1010 to $1B7C). You will see that the prompts table is one contiguous block from $1B7B to $1D29; $1D2A - $1D77 is the message start address list; then another contiguous block from $1D78 to $1DC3 contains three consecutive jump tables: one for the Main Menu, one for the normal mode EDIT HEX keystroke commands, and one for the C/MODE EDIT HEX keystroke commands. Last comes a cursor position table, a table for decimal conversions, then the U search mode table, and space devoted to names you assign when storing programs with devices. This organization facilitates remapping. One needs only use the VICMON N function twice, once with the "W" on the jump tables, and once without the W on the program body to make it work where you want it. In fact, most 6502 code is organized this way. You will, for instance, find VICMON's jump tables starting at its base address + $E80 and ending at its base address + $EB4 if you apply VICMON to itself.

We encourage you to remap Hexkit to where you want it. Use Commodore's VICMON program as follows:

1) Transfer the code with VICMON by starting VICMON with the usual SYS command. Then, enable the alternate page zero at a safe location, and give VICMON the command:

   T 1000 1DFF AAAA <RETURN>

   Where AAAA is the start address in Hex of where you want the Hexkit program to run. You can only move Hexkit by some multiple of 256 bytes. If the move is <u>not</u> some multiple of 256 bytes, the program will not work.

2) Remap the code part of Hexkit with .the VICMON command:

.N AAAA BBBB 0000 1010 1DFF <RETURN>

Where BBBB = AAAA + B69.

3) Remap the jump tables part of Hexkit with the VICMON command:

.N CCCC DDDD 0000 1010 1DFF W <RETURN>

Where CCCC = AAAA + $D1A, and DDDD = AAAA + $DAC.

4) Change the setup pointer for the NAME Kernal subroutine. Use Hexkit's own Q Search to find the call to $FFBD, the call to SETNAM and edit the preload of X and Y as required. This should point to the $DE0th byte of Hexkit where you have put it.

5) Run the Hexkit at its new location by typing SYS (decimal equivalent of AAAA).

****

REMEMBER: THE PROMQUEEN RAM AND EPROM OCCUPY DIFFERENT LOCATIONS IN THE VIC

The PROMQUEEN CARTRIDGE occupies a total of 8 kilobytes of address space (8192 bytes). The first 4 kilobytes (4096 addresses) are allocated to PROMQUEEN RAM. The second 4 kilobytes are allocated to EPROMs. You can easily place the cartridge into any of the 4 VIC expansion blocks.

AS FAR AS THE VIC IS CONCERNED, THE PROMQUEEN RAM AND THE PROMQUEEN EPROM SOCKET NEVER OCCUPY THE SAME ADDRESS SPACE. This means that if you simply burn a machine language program onto EPROM after developing it and debugging it in PROMQUEEN RAM it will not work directly from the EPROM when you attempt to run it from the VIC if any of the following statements are true:

1) Your program calls subroutines inside the PROMQUEEN RAM address space.

2) Your program jumps to addresses within the PROMQUEEN RAM address space.

3) Your program executes indirect jumps to locations within the PROMQUEEN RAM address space.

You must remap any such statements by adding decimal 4096 ($1000) to their operands before you burn the program on EPROM if you want the program to work directly off the EPROM. Record the addresses where such statements occur to make things easy. Alternatively, if you have VICMON be sure to use the "N" routine before you make your burn.

No remapping is necessary for burning EPROMs with programs developed for and tested on external computers with the MIMIC mode.

****

4.3 PREVENTING CRASHES WHEN RUNNING VICMON AND HEXKIT TOGETHER

Hexkit and VICMON are not quite compatible because of the need for both to operate without damaging BASIC programs. You must use the PROMQUEEN reset button to stop VICMON if you don't want to leave the seeds of a crash, but this unfortunately clears any BASIC pointer settings if you are also running a BASIC program.

The alternate crash prevention routine below works without damaging the BASIC pointers. It is a routine that resets the BRK pointer at location $316 - $317 to its default value of $FED2. Just enter this program somewhere and run it after you have exited VICMON with its X command. The routine can be put anywhere without remapping, and is called by the BASIC command SYS (decimal address of the routine). The routine is as follows:

```
SEI             78
LDA #$D2        A9 D2
STA #0316       8D 16 03
LDA #$FE        A9 FE
STA $0317       8D 17 03
CLI             58
RTS             60

     =13 bytes
```

Don't run Hexkit until you have run the above routine first.

4.4 THE HEXKIT 1.0 HEADER

Hexkit 1.0 comes with a 512 byte header written in machine code. This header permits you to load and run Hexkit 1.0 off its EPROM from any of the four VIC expansion blocks. It also lets you make auto start/auto run/auto load--from anywhere--EPROMS out of BASIC programs. If you need to do this in nonstandard situations, the following information will help you. It may be easier to understand if you follow along with your PROMQUEEN set up to operate. Set BLOCK 5 and load Hexkit with SYS 45441. Leave the Hexkit EPROM in and on. Select EDIT HEX and give the address $B000 for the display. Scroll from $B000 up to see the following:

1) In $B000-$B001 is a vector to the start of the user-specified machine code to handle auto starts. $B002-$B003 contain a vector to the SCNKEY Kernal routine. This prevents crashing when you type <RESTORE>.

2) In $B004-$B008 is the sequence of bytes the VIC looks for at address $A004-$A008 during start-up to see if an auto start is requested.

3) In $B00A-$B028 is the code that handles auto starts if it is found at $A00A-$A028. It does the following:

    a) calls the same four Kernal setup routines the VIC usually calls on power up

    b) calls a routine that disables the STOP key

    c) does a little setup and loads the command SYS 41336 <RETURN> into the VIC's keyboard buffer

    d) jumps to the normal BASIC start-up entry.

When the BASIC wakes up and finds that SYS command in the keyboard buffer, it executes it just the same as if you had typed it. The SYS command is the same one you manually type to load and run an EPROMed BASIC program in address block $A000. This particular SYS command calls

the reload to unexpanded VIC's. It must be changed to handle expansions.

4) From $B02A-$B02F is the kill-STOP-key routine called during resets and auto starts.

5) From $B030-$B03F is the keyboard buffer loader routine. This routine uses a vector which has been previously loaded on the VIC's zero page in locations $01-$02 to access a table of character strings to load in the keyboard buffer. The table of character strings begins at $B040 and extends to $B04F. Only part of the table space is used. Special addressing techniques are used to allow this routine with its table to work in the different locations it must. It will work anywhere you want it to provided you first point to its start address in locations $01-$02, and load the Y register with the index which must be added to the start address of the routine to specify which character string to put in the keyboard buffer. Use Hexkit's C/MODE to type in the string just as you would from the keyboard and indicate end of string by using the F3 key. It will figure the number of characters for you, but don't put more than 10 character commands in the keyboard buffer at one time.

6) From $B050-$B080 is a machine code shift routine. It shifts from bottom up only. This routine is fully relocatable, and if used by itself, requires the following setup:

   a) Put the origin of the shift in locations $FB-$FC on the VIC's page zero (low order byte first)

   b) Put the destination of the shift in locations $FD-$FE

   c) Put the number of bytes to be shifted in $0334-$0335

   d) Call the routine.

7) From $B085 to $B005 is a routine which first clears the screen and then proceeds to manage the transfers necessary to load BASIC programs from EPROMs. This is done to enable the same routine to work regardless of its location. The routine pulls the return address of the location from where it was called to find out its origin. From this it calculates the location of the data specifying the setup for the shift routine and what address to use to call the shift and keyboard buffer loader routines. You are giving the information this routine needs to figure out where it is when you type the particular SYS command to load a BASIC program from a particular socket. Once the routine knows where it is, it first looks up from a table pointed at by the pulled return address to do the setup and shift of the BASIC program from EPROM to VIC RAM. The BASIC pointers are also set on the VIC's page zero as required to make the BASIC work. (These are automatically stored by SYS 45312.)

8) At $BODO to $B0FC is the routine that puts the VIC's screen back where it is in an unexpanded VIC. It is called automatically when you use the SYS commands to load to an unexpanded VIC. (This lets Hexkit load properly into expanded VICs.)

9) At $B111-$B119 is the routine you call as SYS 45312 to save BASIC on EPROMs. It does all setup required to prepare the PROMQUEEN RAM with the image of the EPROM to be burned. It does so by doing three shifts. First, it copies the Hexkit header to PROMQUEEN RAM. It then saves the BASIC pointers. Last, it copies the first 3583 bytes from the start of BASIC to the PROMQUEEN RAM starting at $A201. You don't need to do anything special if the VIC is expanded.

10) Then come a series of tables holding setup data. These will be explained in more detail because you may want to doctor them under certain circumstances.

11) After some open space, you come to $B1F0 where the

list of BASIC pointers is stored.

4.5 DOCTORING THE AUTO SAVE HEADER FOR NONSTANDARD RELOADS:

You never have to modify the automatic save routine (SYS 45312) regardless where the BASIC is located and irrespective of the length of the program you want to burn on the EPROM. You do, however, have to modify the reload tables if:

1) Your program is expected to run in a 3K expander.

2) Your program is longer than 3583 bytes and hence will require a second EPROM to hold it all.

The reload program derives control information from a number of tables that tell it

1) the length of the program.

2) where the program is to be loaded to.

The reload program finds the data applicable for the situation by determining from where it was called.

There are 16 reload entry points which allow reloads from the 8 possible addresses into which you can plug EPROMs. Eight of these entry points are for loading to the unexpanded VIC, and because they are used to load Hexkit into VICs which may be expanded, they all first call the routine to RESET the screen to $1E00 so that you don't have to worry about where the VIC screen is when you want to load Hexkit.

Eight more reload entry points allow the reload to occur without change of screen memory location occuring first. These reload entries are for loading to 8K or more RAM expanded VICs.

The table on the following page tells you the SYS command to use to access the reload header properly from wherever you choose to plug in an EPROM. It also tells you where that particular entry point will be found in

the PROMQUEEN RAM after you have used SYS 45312 to do the set up for you.

Let us assume that we have an 4095 byte long program that we want to auto start in an 8K RAM expanded VIC. Here is how to proceed:

1) Load and run the program with the PROMQUEEN in BLOCK 5.

2) Stop it and use the AUTO SAVE (SYS 45312) from your Hexkit EPROM.

3) Look up on the table on page 72 to find where the reload entry for reload to an expanded VIC from BLOCK 5.0 appears in PROMQUEEN RAM after use of SYS 45312. You will find the applicable address is $A1A8.

4) Load Hexkit off its EPROM with SYS 45441. Select EDIT HEX and go to $A1A8. You will see the entry starts with a 6502 JSR A085 command. Scroll down to $A1AB where you will find the low order then the high order byte of the program length. Change these to the new program length + 1, which in Hex is $1000. Remember--low order byte first.

5) The third byte in the table is correct already unless you had wanted the program to load to a 3K expander. If this is the case, change the 12 to a 04.

6) Make sure the auto start SYS command goes to this routine. You will find this character string in $A040 to $A048. Use the Hexkit C/MODE to change this from its present 41336 to 41384. This will cause the auto start to access the entry you just doctored.

7) Burn a 4K EPROM with the present contents of PROMQUEEN RAM. (Burn from $B000 to $BFFF).

8) Reload the BASIC program, run it, then stop it.

9) In <u>Immediate Mode</u> POKE the balance of the program into PROMQUEEN RAM. In our example situation we already EPROMed the first 3583 bytes, so we need only to do the rest. This is

```
 4095
-3583
------
 1512
```

The command to do this is

```
FORI=0 to 1511:POKE 40960+I, PEEK (8192+I):
NEXT
```

Be prepared to wait: BASIC is much slower than Hexkit.

10) Load Hexkit. Put in a fresh EPROM and BURN from $B000 up to the end of the program segment in PROMQUEEN RAM.

11) Use the first EPROM in an expansion socket at $A000 and put the second one in what we call BLOCK 5.5 which is a socket at $B000.

Take some time to study carefully how the reload header works and you will find that you can handle almost any situation.

BY WAY OF REVIEW:

To set up program in VIC for burning on EPROM

A. Install PROMQUEEN in BLOCK 5.

B. Install expansion memory if used.

C. Turn on the VIC and load the program to be EPROMed.

D. With Hexkit EPROM in PROMQUEEN ZIF socket enabled type

SYS 45312 <RETURN>

E. Load Hexkit into VIC with: SYS 45441 <RETURN>

F. Doctor reload header tables as required if:

1) The program is longer than 3583 bytes and/or

2) The program is to be reloaded to a VIC with a 3K memory expander (BASIC to start at $0400).

G. Burn a fresh EPROM with present contents of PROMQUEEN RAM. Burn must be at least decimal 512 + (length of program) bytes long.

H. If program longer than 3583 bytes, reload it in VIC and poke the rest of it into PROMQUEEN RAM starting at 40960.

I. Burn the second EPROM using Hexkit. Install it in VIC address space immediately above first EPROM.

The order of reload table data is: (low byte of program length), (high byte of program length), (high byte of start of BASIC).

4.6 HEXKIT 1.0 AUTO BASIC RELOAD HEADER CALLS EPROM TO VIC

BLOCK	(1) UNEXPANDED VIC	LOC. OF TABLE	(2) EXPANDED VIC	LOC. OF TABLE
1.0	SYS8514	$A142		
1.5	SYS12619	$A14B	SYS12682	$A18A
2.0	SYS16724	$A154	SYS16784	$A190
2.5	SYS20829	$A15D	SYS20886	$A196
3.0	SYS24934	$A166	SYS24988	$A19C
3.5	SYS29039	$A16F	SYS29090	$A1A2
5.0	SYSS41336	$A178	SYS41384	$A1A8
5.5	SYS45441	$A181	SYS45486	$A1AE

1) These calls all first call a routine that resets VIC screen to $1E00.

2) Use these calls if 8K or larger RAM expanded VIC. (Doctor the start of BASIC byte if you must load to a 3K expander.)

5.1 PROMQUEEN CARTRIDGE EPROM COMPATIBILITY

The PROMQUEEN CARTRIDGE is shipped set for proper operation with 2716 EPROMS when its selector switch is set for 2716s, and for proper operation with 2732s with the selector switch set for 2732s.

The programming voltage for 2716s is 25 volts. The programming voltage for 2732s is also 25 volts. There are separate potentiometers accessible through the small holes in the left side of the cartridge for adjusting these voltages. 2732A EPROMs require 21 volts for programming. You can reset one of the pots to get the 21 volts required for 2732As when the selector switch is set for 2732As. See page 51. You can always read a 2732 with the programming voltage set to 21 volts for 2732As. DON'T TRY TO PROGRAM A 2732A WHEN YOU HAVE THE VOLTAGE SET TO 25 VOLTS FOR 2732s. YOU WILL DAMAGE THE 2732A.

Irrespective of EPROM type, the EPROMs you use must be rated for access time of 450 nano-seconds or faster. The 2 megahertz clock of the VIC does not allow sufficient time to read EPROMs which require more than 450 nano-seconds to access. Some 450 nano-second EPROMs are marginal. Use 390 nano-second EPROMs to be safe it you can. Unless you know how to make the proper adaptions NEVER USE ANY OTHER THAN THE 2716, 2732, OR 2732A EPROMs ON YOUR PROMQUEEN. To aid those who want to program 2532's or 2764's with a PROMQUEEN, data sheets for these EPROMs are included on the following pages. You can hot wire for them (but be careful).

2716 EPROMs contain 2 kilobytes of address space per chip (that is 2048 bytes exactly). Each address is uniquely defined by its own combination of 1's and 0's on the eleven address selector lines coming from the chip. Addresses start at binary 00000000000 (eleven zeros) and end at binary 11111111111 (eleven ones).

2732 EPROMs contain 4 kilobytes of address space per chip (that is 4096 bytes exactly). Each address is uniquely defined by its own combination of 1's and 0's on the twelve address selector lines coming from the chip. Addresses start at 000000000000 (12 zeros) and end at 111111111111 (twelve ones).

Conflict between memory devices which could be triggered by identical combinations of 1's and 0's is avoided because each chip requires a signal on an additional pin, called "chip select" in order for it to respond to an address on its address pins. There is a separate chip select line for every memory chip in the VIC. When you change the block setting on the BLOCK selector DIP switch in the PROMQUEEN CARTRIDGE, you are changing which of the VIC's expansion "chip select" lines is connected to the cartridge.

EPROMs contain the number 255 in decimal (FF in hex or 11111111 in binary) in each address as they are shipped from the factory or erased. An ultraviolet lamp is used to erase EPROMs. Such a lamp is available from Logical Devices Inc., among others. The Logical Devices Eraser is the most economical we know of. It costs $49.95. Call Fort Lauderdale, Florida information, then the company to order one. This particular lamp takes 15 minutes to erase an EPROM. Your PROMQUEEN dealer may carry it. He may also cary EPROMs and jumper cables.

5.2 PROMQUEEN ELECTRICAL SUMMARY

1) The address lines enter the cartridge through latching tri-state buffers. These buffers usually run unlatched, unless a low signal is applied to the latch pins. The output of a one-shot multivibrator is connected to the latch pins of the address buffers.

2) This one-shot multivibrator is logically connected so that it triggers when the following are true simultaneously:

   a) The VIC microprocessor read/write line goes low, indicating a write.

   b) The block select line goes low, indicating that the VIC microprocessor is selecting the cartridge.

   c) Address line A11 goes high, indicating that the EPROM is selected.

   d) Triggering is enabled by the setting of the PROG/BURN switch to BURN.

3) When the one-shot triggers, the EPROM address to which the write was attempted is frozen on the cartridge by the latching of the address buffers; address lines A0 to A10 are decoded both to the EPROM and to the PROMQUEEN RAM chips (2-6116's). The chip enable line to the 6116 selected by A10 is brought low, causing the data in the selected RAM chip to be output onto the PROMQUEEN's internal data bus. The VIC's data bus was disconnected from the PROMQUEEN's data bus when the cartridge was set to PROG, thus the data written by the VIC never gets to the EPROM. Instead, the data from that single selected RAM address is applied to the EPROM data pins through a buffer. Since the address is held stable by the latches, the data stays stable, too. Meanwhile the same set of signals on address lines A0-A11 is

being applied through buffers to the EPROM's address pins. The programming voltage is also being applied; this started when PROG/BURN was set to BURN.

4) A few microseconds after the one-shot triggers, its output causes a second one-shot multivibrator to trigger. This second one-shot is timed to remain triggered for 50 milliseconds to generate the programming pulse to the EPROM. The programming pulse is applied to the EPROM's chip enable pin.

5) After the conclusion of the programming pulse, the first one-shot untriggers which once again unlatches the address buffers. The cartridge can now look for a new attempted burn to a location on EPROM.

REMEMBER it takes about 70 milliseconds to burn a single EPROM location. This is why you need to generate a time delay between each burn signal to the PROMQUEEN cartridge.

Note the following, if you are interested in the electronics:

a) That both the data lines and the address lines connected between the EPROM socket and the PROMQUEEN RAM pass through tri-state bus transceivers (74LS245's). You are putting these buffers into high impedance and turning off both Vcc and Vpp to the EPROM when you turn the SOCKET ON/OFF switch to off.

b) That the VIC's data lines pass through a buffer (another 74LS245) before connecting to the PROMQUEEN RAM chips. This buffer is bidirectional, with its direction under control of the VIC's R/W line. It is also tri-statable, and remains at high impedance unless the PROMQUEEN cartridge address space is selected by the VIC's BLOCK signal. Additional logic provides that

this buffer remains in the high impedance condition if the cartridge is set to BURN mode, or if the MIMIC mode is selected.

c) That the address latches buffering the VIC's address bus into the PROMQUEEN CARTRIDGE are tri-statable. You are placing these buffers into high impedance when you select MIMIC mode.

d) That the buffers on the address lines to the EPROM socket are bidirectional. You are setting them so that address information is passed from PROMQUEEN RAM to the EPROM socket when in PROG or BURN mode. When you set the MIMIC mode, these buffers turn around to pass address codes received from jumper cables in the EPROM socket through to the PROMQUEEN RAM.

e) That the data buffer between the PROMQUEEN RAM and the EPROM socket, being bidirectional, is set to pass information out from PROMQUEEN RAM to the EPROM socket when in BURN or MIMIC mode. This buffer passes data from the EPROM socket on through to the VIC only when the cartridge is set to PROG mode.